# Combating Tracking Drift: Developing Robust Object Tracking Methods

**Jiatong Li**

Faculty of Engineering and Information Technology

University of Technology, Sydney

This thesis is submitted for the degree of

*Doctor of Philosophy*

2017

# Certificate of Original Authorship

I certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This thesis is the result of the research candidature conducted jointly with Beijing Institute of Technology as part of a collaborative doctoral degree.

Signature of Student: ____________

Date: 26/7/2017

I would like to dedicate this thesis to my loving parents and wife

# Acknowledgements

I would like to express my sincere appreciations to those who have supported me and contributed to this thesis during my PhD journey.

First of all, I would like to thank my supervisor, Prof. Richard Xu, for his valuable guidance and useful discussions during my research developing stage. He is who led me to the field of academic research, and who gave me a great opportunity to broaden my research view. His passion for academic knowledge always motivates me to move forward towards my research goals. Besides, I wish to show great gratitude to my co-supervisor Prof. Massimo Piccardi, and Prof. Min Xu, Prof. Qiang Wu for their great support during the course of my study. I would also like to thank Prof. Baojun Zhao and Prof. Chenwei Deng for their great help when I was in Beijing Institute of Technology. Very special thanks to Prof. Dacheng Tao for his invaluable insights and suggestions for the last stage of my PhD work.

Heartfelt thanks to my fellow labmates in Faculty of Engineering and Information Technology, UTS. Also great thanks to my friends and collaborators. We worked and enjoyed the life together for our intensive and joyful time. They are: Minqi Li, Xiang Feng, Shuai Jiang, Chang Liu, Tianrong Rao, Qingbo Xia, Lingxiang Wu, Tian Ding, Bin Yang, Peibo Duan, Felix, Sheng Wang, Zhichao Sheng, Maoying Qiao, Junyu Xuan, Wenjie Zha, Hongshu Chen, Fan Dong, Haimin Zhang, Baosheng Yu, Tongliang Liu, Zhe Chen, Zijing Chen. In particular, I am grateful to Jessica Anne Washington for her help in the thesis writing.

Finally, I would like to show my deeply gratitude to my family: my parents and my wife, for always supporting me selflessly in my life and study. It's their constant encouragement and determination that made all these possible ultimately.

# Abstract

Visual object tracking plays an important role in many computer vision applications, such as video surveillance, unmanned aerial vehicle image processing, human computer interaction and automatic control. This research aims to develop robust object tracking methods, which are capable of tracking general object without the prior knowledge of the target. Tracker drift is one of the most challenging issues in object tracking due to target deformations, illumination variations, abrupt motions, occlusions and background clutters. This thesis focuses on the tracking drift problem, and adopts three main solutions. These include: designing an efficient target shape feature extraction method, comparing target features with metric learning and using the ensemble tracking method to tackle the tracking drift during tracker online update. The main work and contributions are as follows:

1. We propose a Weber's Law Shape Descriptor (WLSD) for efficient object tracking under background clutters. Weber's Law indicates that the perceived change in stimuli is proportional to the initial stimuli, which means that the saliency variations is not only relative to the feature variations, but also the initial feature quantity. Motivated by Weber's Law, this thesis proposes a Weber's Law Shape Descriptor to describe the saliency variations of the shape contour. The proposed method first designs a Contour Angular Feature as the initial stimuli, and builds the WLSD according to Weber's Law. Then, WLSD is extended to multi-scale to enhance its shape discriminative strength. Finally, a feature selection scheme is used to extract the effective WLSD scales. The proposed WLSD is naturally invariant to the shape scale and rotation, and has low computation load as a whole shape descriptor. We further apply WLSD in thermal infrared object tracking, and propose a multi-feature integration method based on WLSD shape, target area and target trajectory. The experiments are first conducted on MPEG-7 and Tari shape dataset to test the shape discriminative capacity of WLSD, and then tested on the infrared tracking videos to validate the proposed multi-feature integration tracking framework.
2. We propose a Time Varying Metric Learning (TVML) object tracking method. Recently, tracking-by-detection (TBD) methods are very popular. Traditional TBD methods track the objects by training a binary classifier to discriminate the target and background, which

leads to two main issues: firstly, the classifier is unreliable when trained with insufficient data, secondly, comparing the object features with traditional Euclidean distance leans to tracker drift. To solve the above problems, we propose a time varying metric learning model and apply it to object tracking. The proposed TVML adopts Wishart Process to model the variation of the positive semi-definite (PSD) matrices, i.e. the metrics, and uses the Recursive Bayesian Estimation (RBE) framework to learn the metrics under the side information constraint. We introduce the side information constraint to omit the clustering of the negative samples, which is very suitable to the background cluster tracking scenarios. Furthermore, the RBE framework guarantees the proposed model is able to estimate the metrics with limited training data. The experimental results demonstrate the comparable performance of the TVML tracker compared to many state-of-the-art methods on OTB-50 dataset.

3. We propose a historical tracker snapshots based ensemble tracking method. There are frequent target appearance changes due to illumination variations, abrupt motions and target deformations, which needs the tracker to conduct online update to adapt to the target appearance variations. The online update of the tracker often leads to drift. This thesis proposes a historical tracker snapshots based ensemble tracking framework, and designs a Scale-adaptive Multi-Expert (SME) tracker according to the proposed method. The tracker ensemble is composed of the current tracker and its historical tracker snapshots. When the tracker drift is detected, the framework will select the suitable tracker snapshot to replace the current drift tracker according to their accumulated scores. Due to the fact that the tracker tends to be more confident to its own prediction, we propose to define the tracker score in a semi-supervised learning perspective to describe the consistency and the ambiguity of the tracker ensemble simultaneously. We use the regression correlation filter as the base tracker due to its high efficiency. Furthermore, we propose to establish the target scale pyramid to estimate the target scale accurately. The proposed SME tracker is tested on the OTB-50 and VOT2015 tracking dataset, which demonstrates the excellent performance of the proposed tracker with real time speed.

4. We propose a discrete graph based ensemble tracking method. The tracker ensemble is constituted by the current tracker and its historical snapshots as the multi-expert. This thesis proposes to introduce the discrete graph to model the tracker ensemble, where the graph node represents the expert hypothesis. After defining the unary and pair-wise score of the graph, the best expert is selected according to the graph path of the highest score. With the efficient solver of dynamic programming, the proposed method can implicitly analyze the reliability of the multi-expert trajectories by only computing their scores in the current frame, so as to correct the tracker drift. We integrate three base trackers into the proposed

tracking framework to validate its generality, including online support vector machine on a budget, hand-craft feature based correlation filter and convolutional neural network based correlation filter. The proposed three trackers are widely tested on the OTB-50, OTB-100 and VOT2015 dataset, which demonstrates the proposed trackers are superior to the compared state-of-the-art trackers in both the tracking accuracy and robustness measures.

Keywords — Object tracking, Weber's Law, time varying metric learning, ensemble tracking, correlation filter, discrete graph.

# Table of contents

# List of figures

# List of tables

# Chapter 1

# Introduction

## 1.1 Research Background

### 1.1.1 Image Processing and Computer Vision

Visual system is one of the most important perceptual systems to process visual detail. Image Processing is a technique for processing images using mathematical operations. By using any form of signal processing algorithms, the input of an image processing system can be an image or a video, and the output may be either an image or a set of characteristics or parameters related to the image. Image processing is one of the important branches in signal processing that involves treating the image as a two or three dimensional signal and applying standard signal processing techniques to it. Computer vision is closely related to image processing, which is considered as high-level image processing aiming to “teach” computers to decipher the physical contents of an image, or even predict, learn and infer.

The visible light is the portion of the electromagnetic spectrum that is visible to human visual system, with wavelengths from 400 to 700 nanometres. Additionally, infrared thermal image processing has also attracted much attention in recent years. This thesis limits to the visible and infrared image processing that is actively researched in computer vision community.

Computer vision technique requires high computation capacity for data processing machines. Not until 1970s that with the development of computer performance had computer vision began to attract much attention. Currently, in the era of internet and big data, thanks to the rapid growing of techniques including large scale integrated circuits, parallel and distributed computing, algorithms that need high computation load have been revived. One of the representative technique is deep learning, which has been made revolutionary changes in computer vision field with the help of big data and graphics processing unit [1]. Now computer vision has been used in many applications, for instance, intelligent transport, video surveillance, automatic driving, robotics, medical image analysis and human machine interaction.

### 1.1.2 Object Tracking

Object tracking is a typical application in the field of computer vision, which plays a key role in many practical applications, such as video surveillance, intelligent transport, human computer interaction and artificial robots. The rapid development of computer vision and machine learning has seen the robust object tracking system attracting more interest from academics and industries, as an inexpensive and quality solution. Although there has been a lot progress in the object tracking field both in accuracy and robustness, challenging problems still remain due to illumination variations, rotations, distractors, abrupt motions, occlusions and background clutters, which are the major factors for tracking drift ([2, 3, 4, 5]).

The aim of object tracking is to robustly estimate the position (or scale) of the object of interest successively in the video sequences. There are typically two branches of the tracking scenarios. One is assumed that the prior knowledge of the object is available before the tracking procedure begins ([6]), another requires no prior knowledge, and all the tracking process is based on the limited information provided by the initialization of the target at the beginning of the video. In the latter scenario, since the algorithm is requested to track any arbitrary object, it is impossible to get gathered data and train a specific detector, which is more challenging ([7, 8, 9, 10]). Furthermore, the target appearance may vary all the time, therefore the tracker should update incrementally on-the-fly to adapt to the object variation. Our research is focused on the latter scenario. This problem is a challenging task due to target deformations, illumination variations, abrupt motions, partial occlusions and background clutters, which can cause tracker drift or even tracking failure subsequently.

## 1.2 Literature Review

Object tracking has been extensively studied over the past twenty years ([3, 4, 5]) and significant progress has been made. Recent public available benchmark and evaluation have also accelerated the development of this field ([2, 3, 11]). Although numerous methods have been proposed, they can be roughly divided into two categories, i.e. generative methods and discriminative methods. Recently, discriminative methods play the main role in the tracking community. Therefore, we review mainly the discriminative methods, and among the generative methods, we focus on the classic particle filter. In the discriminative methods, we review the popular tracking-by-detection methods, state-of-the-art correlation filter methods and the ensemble methods which are related to our work.

### 1.2.1 Generative Methods

Generally speaking, the generative methods try to represent the target by the appearance model. In the generative framework, such as subspace learning ([12, 13]) and sparse representation ([14, 15]), tracking can be regarded as searching the most similar candidate as the target. Template models are usually learned and updated to adapt to the target appearance variation. The most popular framework in the generative methods is the particle filter. In the following section, we review the detail in particle filter and its related papers.

Traditional Kalman filter method is a well-known tracking framework that provides the optimal state estimation under linear assumption with Gaussian noise [16]. However, it may not perform well in visual object tracking scenario, as the object often moves with much uncertainty. Particle filter ([17]), also known as Sequential Monte Carlo ([18]) method, is also a recursive Bayesian estimator as Kalman filter. Particle filter provides a convenient way to solve the non-linear/non-Gaussian dynamic system ([19]) that overcomes the limitation of Kalman filter, which is extensively used in the object tracking field as the motion model. Although particle filter is adopted in object tracking since the beginning of 2000 ([20]), much recent work is still based on this framework while to pursue more robust appearance models ([14, 15, 21]). Others use particle filter to estimate the state of target local patch instead of the state of the whole target ([22]). In addition, the Markov Chain Monte Carlo methods are also introduced for target state estimation of higher efficiency ([13, 23, 24]).

Generally, denoting the target state as $\mathbf{x}_t = \{p_x, p_y, \alpha_1, \alpha_2, \alpha_3, \alpha_4\}$, where $\{p_x, p_y\}$ are the target translation and $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$ are the affine transformation parameters. The tracking procedure is modeled as the recursive Bayesian filtering form:

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) \propto p(\mathbf{y}_t|\mathbf{x}_t) \times \int_{\mathbf{x}_{t-1}} p(\mathbf{x}_t|\mathbf{x}_{t-1}) p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}) d\mathbf{x}_{t-1}, \tag{1.1}$$

where $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ is the posterior, $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ and $p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})$ are the likelihood and prior respectively.

In the particle filter framework, the whole procedure is divided into two steps: propose and update. The posterior is represented as a finite set of weighted particles, which can be denoted by $\bar{p}(\mathbf{x}_t|\mathbf{y}_{1:t}) = \sum_{i=1}^{N} w_t^{(i)} \delta_{\bar{\mathbf{x}}_t^{(i)}}(\mathbf{x}_t)$, where the symbol bar denotes the estimation. Then the each new particle is proposed(propagated) by the motion model $p(\mathbf{x}_t|\mathbf{x}_{t-1})$. The update step is conducted by re-weighting each particle according to the likelihood $p(\mathbf{x}_t|\mathbf{x}_{t-1})$, which is usually evaluated by the similarity between the target candidate and the target template. The importance weight of particle $i$ is then denoted by $w_t^{(i)} = w_{t-1}^{(i)} p(\mathbf{y}_t|\bar{\mathbf{x}}_t^{(i)})$. Finally, the resampling step is conducted to avoid degeneracy ([25, 17]).

### 1.2.2 Discriminative Methods

Although generative methods have powerful descriptive capacity and needs small number of training set, they pay most of the attention only to the target and less to the background, which make them hard to adapt to the challenging tracking scenarios such as background clutter or the presence of distracters. Unlike generative methods, discriminative methods treat the tracking task as a binary classification problem, which aims to train online classifier to discriminate target and the background. The very popular idea of tracking-by-detection is mostly based on discriminative methods.

#### Tracking-by-detection Methods

The tracking-by-detection method plays a key role among numerous recent tracking methods. Under this framework, a discriminative classifier is trained to classify the foreground and background features ([6, 26, 8, 27, 10]). Powerful discriminative classifiers are introduced and adapted to the tracking domain, such as Support Vector Machine (SVM) ([6]), boosting ([28, 29]), random forest ([7]) and so on.

SVM is regarded as one of the most popular off-the-shelf classifiers to deal with small set of training data. Avidan [6] integrates the SVM classifier into the optical flow to establish the online target discriminative model. However, the main problem is that the pre-trained SVM makes it not suitable for applying SVT to generic object tracking application. Recently, Hare et al. [10] use structured output SVM and trains samples with structured labels, which shows excellent performance in the benchmark ([2]).

Boosting is another powerful classifier, which adopts a set of weak classifiers to create a strong classifier. It is widely used since its first breakthrough success in face detection ([30]). Grabner et al. [28] propose to employ Adaboost to select discriminative features online to ensemble a strong classifier. However, this methods tends to drift with the continuous update mechanism. Later, Grabner et al. [29] introduces the semi-supervised boosting methods to tackle the drift problem. Unlike the above work, Babenko et al. [8] introduce multiple instance learning to collect positive and negative samples into bags to avoid model drift. The MIL tracker not only takes advantage of the boosting methods in feature selection but also efficiently handles the ambiguous labeling problem.

Besides SVM and boosting, the idea of random forest is also adopted to design the tracker. In [7], random forest is introduced to train a binary classifier with the pair-wise gray scale feature for each base classifier. The proposed random forest is then fed to the nearest neighbor classifier in a cascade manner for rejecting the false positive bounding boxes in high efficiency.

### Correlation Filter Methods

In the last five years, Correlation Filter (CF) based tracking methods have attracted great attention due to its simplicity and high efficiency ([31, 32, 9, 33, 34, 35]). Since Bolme et al. [31] propose a minimum output sum of squared error filter for tracking, correlation filter began to re-attract attention as a commonly used method in signal processing.

Correlation filter methods aims to learn the template filter to predict the target location. The correlation operation between the filter and the target template presents maximum value at the center of the target location while the output degenerates gradually as the location far away from the target center. To some extent, CF tracking is also under the framework of tracking-by-detection, but with high efficiency when evaluated using Fast Fourier Transform. Since CF methods are the newest trend in object tracking, and are closely related to the work in this thesis, we review CF individually in this section.

In the following sections, we first review the single channel Correlation Filter and multiple channel Correlation Filter respectively, and then introduce the recent development and trend in correlation filter.

**Single Channel Correlation Filter.** Correlation is a basic concept in signal processing. Given two one-dimensional signal sequences $x(n)$ and $y(n)$, the definition of linear correlation between signal $x(n)$ and $y(n)$ is:

$$r_{xy}(m) = \sum_{n=-\infty}^{+\infty} x(n)y^*(n-m). \tag{1.2}$$

Generally speaking, signal processing methods are usually operated on finite discrete signals, which leads to the so-called circular correlation, whose definition is as follows:

$$\begin{aligned} r_{xy}(m) &= \sum_{m=0}^{N-1} y^*(n)x((n+m))_N R_N(m) \\ &= \sum_{n=0}^{N-1} x(n)y^*((n-m))_N R_N(m), \end{aligned} \tag{1.3}$$

where $x((n))_N$ is the extended periodic signal with period $N$ of original N points $x(n)$, and $R_N(n)$ is the rectangular sequences:

$$R_N(n) = \begin{cases} 1, & 0 \leq n \leq N-1 \\ 0, & \text{others}, \end{cases} \tag{1.4}$$

which means the output of circular correlation only resides within $0 \leq n \leq N-1$. Correlation Filter is traditionally posed in the frequency domain. However, recent literature suggest that viewing CF in the spatial domain can give us more intuitive insights, especially in the computer vision domain ([9, 36, 37]).

MOSSE correlation filter ([31]) can be expressed in the spatial domain as a ridge regression problem:

$$E(\mathbf{h}) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{D} ||\mathbf{y}_i(j) - \mathbf{h}^T\mathbf{x}_i[\Delta\tau_j]||_2^2 + \frac{\lambda}{2}||\mathbf{h}||_2^2, \tag{1.5}$$

where $\mathbf{y}_i \in \mathbb{R}^D$ is the desired response for the i-th input $\mathbf{x}_i \in \mathbb{R}^D$, while $\mathbf{x}_i[\Delta\tau_j]$ $(j = 1,2,...D)$ represents all the circular shifts for input $\mathbf{x}_i$, and $\mathbf{h}$ is the learned filter. Bolme et al. [31] suggest using a 2D Gaussian with small variance for $\mathbf{y}_i$ which is centered at the location of the object.

It is well known that the circular correlation in the spatial domain is equivalent to the element-wise product in frequency domain [38]. Therefore, the objective of Eq. 1.5 can be expressed as:

$$\begin{aligned} E(\hat{\mathbf{h}}) &= \frac{1}{2}\sum_{i=1}^{N} ||\hat{\mathbf{y}}_i - \hat{\mathbf{x}}_i \odot \text{conj}(\hat{\mathbf{h}})||_2^2 + \frac{\lambda}{2}||\hat{\mathbf{h}}||_2^2 \\ &= \frac{1}{2}\sum_{i=1}^{N} ||\hat{\mathbf{y}}_i - \text{diag}(\hat{\mathbf{x}}_i)\text{conj}(\hat{\mathbf{h}})||_2^2 + \frac{\lambda}{2}||\hat{\mathbf{h}}||_2^2, \end{aligned} \tag{1.6}$$

where the symbol ^ denotes the Fourier transform, and diag() is an operator that reorder a $D$ dimensional vector into a $D \times D$ dimensional diagonal matrix. In addition, the complex conjugate is employed to $\hat{\mathbf{h}}$. By minimizing Eq. 1.6 and taking the gradient, we can get the closed solution:

$$\begin{aligned} \hat{\mathbf{h}^*} &= [\text{diag}(\hat{\mathbf{s}}_{xx}) + \lambda\mathbf{I}]^{-1}\sum_{i=1}^{N}\text{diag}(\hat{\mathbf{x}}_i)\hat{\mathbf{y}}_i \\ &= \hat{\mathbf{s}}_{xy} \odot^{-1} (\hat{\mathbf{s}}_{xy} + \lambda\mathbf{1}) \end{aligned} \tag{1.7}$$

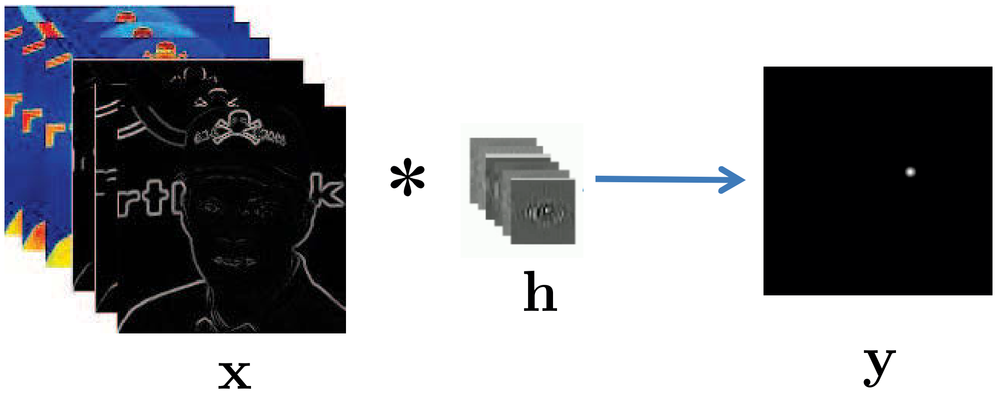


Fig. 1.1 Multiple Channel Correlation Filter Illustration.

where $\odot^{-1}$ denotes element-wise division, and

$$\begin{aligned}\hat{\mathbf{s}}_{xy} &= \sum_{i=1}^{N} \hat{\mathbf{y}}_i \odot \mathrm{conj}(\hat{\mathbf{x}}_i)\\ \hat{\mathbf{s}}_{xx} &= \sum_{i=1}^{N} \hat{\mathbf{x}}_i \odot \mathrm{conj}(\hat{\mathbf{x}}_i)\end{aligned} \tag{1.8}$$

The computation of the filter $\hat{\mathbf{h}}$ in frequency domain can be found with the cost of $\mathcal{O}(ND\log D)$, while its equivalence objective function in time domain has the cost of $\mathcal{O}(D^3 + ND^2)$, which indicates the superior advantage of correlation filter ([37]).

When applied in the tracking field, the input image $\mathbf{x}_i$ is obtained sequentially. Therefore, the MOSSE filter adopts a linear interpolation methods, which can be expressed as follows:

$$\begin{aligned}\hat{\mathbf{h}}_t^* &= \frac{A_t}{B_t}\\ A_t &= \eta \hat{\mathbf{y}}_t \odot \mathrm{conj}(\hat{\mathbf{x}}_t) + (1-\eta)A_{t-1}\\ B_t &= \eta \hat{\mathbf{x}}_t \odot \mathrm{conj}(\hat{\mathbf{x}}_t) + (1-\eta)B_{t-1},\end{aligned} \tag{1.9}$$

where $A_t$ and $B_t$ denote the numerator and denominator of the filter $\hat{\mathbf{h}}_t^*$ respectively, and $\eta$ is called the learning rate.

**Multi-Channel Correlation Filter.** In the previous section, we review the single channel correlation filter. However, it is more robust to adopt CF into the multi-channel feature map, which desires the multi-channel correlation filter naturally. As illustrated in Fig. 1.1, multiple channel CF aims to train multiple channel filters with each channel operates with the corresponding target feature channel, and output the desired output [37].

In tracking scenario, as the training data incrementally increases, we derive the CF using single training data for simplicity. Consider the $K$ dimensional feature map of an image. Let $\mathbf{x}$ denote a rectangular patch extracted from the feature map, which is composed of $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_K$. We aim to find a multi-channel correlation filter $\mathbf{h}$ which consists of $K$ dimension corresponding to each feature dimension. The objective is by minimizing the following cost function:

$$E(\mathbf{h}) = \| \sum_{l=1}^{K} \mathbf{h}^l \otimes \mathbf{x}^l - \mathbf{y} \|_2^2 + \lambda \sum_{i=1}^{K} \|\mathbf{h}^l\|_2^2, \tag{1.10}$$

where $\mathbf{y}$ is the desired correlation output and symbol $\otimes$ denotes the circular correlation operator. Here, $\lambda \geq 0$ is regularization term. As a result, Danelljan et al. [39] take advantage of the multi-channel correlation filter in Eq. 1.10 whose solution can be expressed as:

$$\hat{\mathbf{h}}^l = \frac{\text{conj}(\hat{\mathbf{y}}) \odot \hat{\mathbf{x}}^l}{\sum_{k=1}^{K} \text{conj}(\hat{\mathbf{x}}^k) \odot \hat{\mathbf{x}}^k + \lambda}. \tag{1.11}$$

Like the single channel correlation filter, the numerator and denominator of the filter is updated individually:

$$\begin{aligned} A_t^l &= \eta \text{conj}(\hat{\mathbf{y}}_t) \odot \hat{\mathbf{x}}_t^l + (1-\eta) A_{t-1}^l \\ B_t^l &= \eta \sum_{k=1}^{K} \text{conj}(\hat{\mathbf{x}}_t^k) \odot \hat{\mathbf{x}}_t^k + (1-\eta) B_{t-1}^l, \end{aligned} \tag{1.12}$$

where $\eta$ is the learning rate.

The target location of the new frame is found by computing the correlation between the Fourier transform of the new feature map $Z$ and the learned filter:

$$y = \mathcal{F}^{-1} \left\{ \frac{\sum_{l=1}^{K} \text{conj}(A^l) Z^l}{B + \lambda} \right\} \tag{1.13}$$

**Correlation Filter Trackers and its Latest Trends.** Correlation Filter is very popular and most of the state-of-the-art trackers are the improvement or based on it.

Henriques et al. [32] propose to use circular image patches as dense samples to train the correlation filter in kernel space with low computation load. However, the above methods are based on gray-level feature. The work is further extended to HOG feature in the Kernelized Correlation Filter (KCF) tracking algorithm [9]. In [33], color attributes are added to the framework of [32], and an adaptive dimension reduction technique is proposed, which demonstrates the importance of color feature in object tracking. Motivated by TLD tracker [7], Ma et al. [35] add the re-detection scheme to CF to achieve long term correlation tracking. In [34],

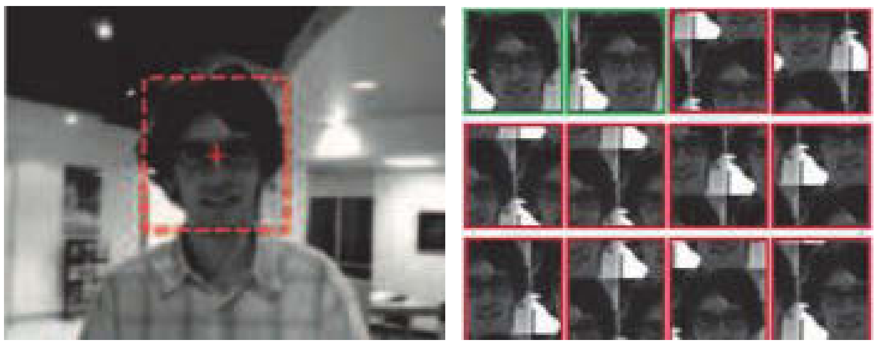


Fig. 1.2 Boundary effect of Correlation Filter. (a) Defines the example of fixed spatial support within the image from which the peak correlation output at the center of the bounding box. (b) A subset of patch examples used in a correlation filter where green denotes a non-zero correlation output, and red denotes a zero correlation output.

accurate scale estimation is obtained by treating translation and scale correlation separately, and a one-dimensional scale correlation filter is used to measure scale change.

Correlation filter takes advantage of FFT, which is the fast algorithm of Discrete Fourier Transform (DFT). However, the circular correlation assumption in DFT induces the image boundary effect, which is considered to be detrimental. As shown in Fig. 1.2, with the exception of few samples near the center of the input image patch, most of the circular shifts of the image patch are plagued by circular boundary effects and are not truly representative of the normal shifted examples. According to the relationship between the linear correlation and circular correlation, the boundary effect can be reduced by augmenting with zeros. However, expanding in the spatial domain destroys the computation efficiency. To resolve the boundary effect, Galoogahi et al. [36] propose to add mask matrix containing ones and zeros that encapsulates which part of the signal should be active/inactive, so as to increase the proportion of examples unaffected by boundary effect dramatically. On the contrary, in [40], Danelljan et al. propose to add the spatial regularization component to further suppress the background region.

From 2015, Correlation Filter has been commonly used as an accurate and efficient object location framework and is integrated with more advanced feature. For example, in [41, 42, 43], Convolutional Neural Network (CNN) feature is integrated into correlation filter framework, and achieves more advanced performance, especially for non-rigid target, which is hard to handle in the traditional hand-craft feature (like HOG and SIFT feature).

### Convolutional Neural Network Methods

Because of the powerful feature representation capacity of convolutional neural network (CNN), the CNN model has been adopted into visual tracking recently. These CNN based methods are usually designed by integrating the extracted feature to distinguish target appearance from the background with the grid searching or the advanced correlation filter searching framework. Since the excellent transfer property of the pre-trained CNN model on large scale image dataset [44], many trackers use the pre-trained CNN model directly as the feature extractor [45, 41, 46, 47]. Otherwise, in [48], the authors train a specific CNN model for tracking in a multi-domain manner which won the VOT2015 challenge [11]. In [49], the tracking algorithm employs multiple CNNs collaborate to estimate the target states by maintaining them in a tree structure. Recently, Fan and Ling [50] propose a structure-aware network to incorporate recurrent neural network into CNN to improve the tracking robustness in presence of similar distractors.

Recent two years, Siamese CNN model is introduced into tracking community to discriminate whether two image patterns include the same object. In [51], a siamese neural network is trained offline to learn the matching function. Once learned, the network does the tracking job simply by matching the initial patch of the target to the candidates without online adapting, which still reaches good performance. The similar idea can be found in [52]. The above methods bypass the traditional online learning strategy. However, only using a fixed metric learned offline to compare target appearance prevents the tracker from exploiting the sequence specific cues. Therefore, in [53], Valmadre et al. propose an end-to-end training framework to learn deep features that are tightly coupled to the correlation filter, which has made a good trade-off between the low speed of SGD training based CNN model and the high speed of correlation filter.

### 1.2.3 Ensemble Methods

Some tracking algorithms adopt tracker ensemble to achieve more robust tracking performance. In this thesis, we call these algorithms ensemble tracking methods. For example, Kwon and Lee [13] decomposes traditional Bayesian recursive framework into basic models, and uses the MCMC sampling to integrate them. In [23], the proposed method samples the target state space as well as the tracker space to handle challenging tracking scenes. Kalal et al. [7] address the long term tracking problem by designing two complementary experts, one estimates missed detections and the other estimates false alarm, apart from this, a re-detection scheme is designed to achieve long term tracking. In [15], a sparsity-based collaboration of discriminative and generative modules are proposed. Hong et al. [54] adopt the hierarchical appearance model to track object through multi-level. In ([55]), multiple experts are used to handle the model drift problem, which shows high tracking efficiency.

In addition, there are some trackers aim to model the advanced base tracker ensemble into a unified framework to further boost the tracking performance. For instance, in [22], the reliable patch is exploited within the particle filter framework, while correlation filter [9] is used as the base tracker. Unlike the traditional methods which use particle filter to estimate the target state, in [22], each particle represents the state of the local patch on the target. The proposed tracker then aims to propagate the reliable patch on the target as long as possible. In [21], the proposed ensemble tracking method is used to deal with the part based object, and different parts are integrated within the Bayesian framework. Recently, in [49], The CNN based tracker ensemble is modeled by a tree structure and update online along the path in the tree, which is capable of capturing diverse target appearances.

## 1.3 Tracking Drift

Object tracking has been actively researched for more than 20 years. Despite significant progress, object tracking still faces challenges from internal and external factors, such as target deformation, illumination changes, motion blurs, target occlusions and background clutters etc [2, 3]. All the above challenging scenarios can lead to tracker drift, which is commonly referred when a tracker deviates from the true target location gradually, and finally fails to tracker. The factors that lead to tracking drift can be roughly divided into two categories:

- Target Appearance Variation.

  The reasons for the target appearance changes can be summarized into internal factors and external factors. The internal factors are referred to target appearance variations caused by the changes of the target itself, including object rigid or non-rigid deformations, and scale changes. While the external factors are referred to outer factors, like illumination changes, target occlusions, camera motion blur and low resolution etc. To tackle the above issues, an online target model is usually maintained, which updates itself to adapt to target appearance changes. On the other hand, robust target representation is capable of simplifying the tracker design.

- Background Clutter.

  Another main reason for tracker drift is the background clutter. Note that background clutter does not mean the complex degree of the background, but the similarity of the target and background. If the target is distinct from background, even the tracking environment is full of complex objects, we can also track the target successfully. On the contrary, if there are objects very similar to the target in the background, the tracker usually fails to track easily. To track under complex background, it's suitable to use discriminative methods to distinguish the object from background. Highly discriminative target representation is also preferred in this scenario. Another idea is to learn an appropriate metric instead of traditional Euclidean distance to better distinct object features from the background.

In summary, tackling tracker drift problem is equivalent to handling target appearance variations and background clutters. In this thesis, we focus on three aspects to tackle the above two issues: target representation, metric learning and object online model update, and propose a series of tracking drift suppression methods.

## 1.4 Contributions

In this thesis, we aim to combat the problem of tracking drift by raising three issues:

- To design a more efficient target feature extraction method
- To find a more robust feature matching method
- To establish a more effective online model update scheme

We propose three solutions to achieve these aims, including target shape feature extraction, distance metric learning based feature matching, and ensemble tracking based method to suppress tracker drift. The main contributions are as follows:

1. In Chapter 2, we propose a Weber's Law Shape Descriptor (WLSD), and apply the proposed method into infrared object tracking.

   Motivated by Weber's Law from psychology, we propose a WLSD shape feature extraction method to describe the saliency variations of the shape contour. We first propose a Contour Angular Feature (CAF) as the base feature, and then establish WLSD based on CAF according to Weber's Law. To extend the capacity of WLSD, we extend it to multi-scale, and adopt feature selection method to select discriminative scales. WLSD is naturally invariant to shape rotation and scale changes. Moreover, as a global shape descriptor, WLSD has much less computation load than local shape descriptors. We further apply the proposed WLSD into infrared object tracking, and propose a multi-feature integration tracking method including target shape, area and motion trajectory. In the experiments, we first test WLSD on two publicly available shape datasets, MPEG-7 and Tari, and then validate the proposed tracking method in infrared videos.

2. In Chapter 3, we propose a Time Varying Metric Learning (TVML) model for object tracking, in order to better distinguish the target and background.

   Traditional tracking-by-detection method trains the binary classifier to discriminate the target from background, which is prone to drift when the quantity of training data is not large enough. On the other hand, sometimes, using Euclidean distance to compare feature vectors leads to false matching. To tackle the above issues, we propose a TVML model for object tracking. The proposed TVML is under Recursive Bayesian framework, which can estimate the metrics with limited training data. In addition, we introduce "side information" constraint to the training data instead of single label constraint, which is capable of omitting the negative clusters, and is more suitable for background clutter tracking scenarios. The experiments on OTB-50 dataset demonstrate the effectiveness of the proposed method.

3. In Chapter 4, a tracker snapshot based multi-expert tracking method is proposed to tackle the drift for tracker update.

   The multi-expert framework is composed of the current tracker and the tracker snapshots. When the tracker drift is detected, the multi-expert framework is capable of selecting the reliable tracker snapshot to correct the drift by replacing the current tracker. As the single expert tends to be more confident to its own prediction, we propose to define the expert score in the semi-supervised manner, which is able to describe the multi-expert consistency and ambiguity in a unified formulation. Furthermore, by integrating correlation filter as the base tracker, and the target scale estimation scheme, we propose a Scale-adaptive Multi-Expert (SME) tracker. SME is extensively evaluated on OTB-50 and VOT2015 dataset, which demonstrate its excellent performance against many state-of-the-art trackers with real-time speed.

4. In Chapter 5, we extend the work of Chapter 4, and propose a discrete graph based multi-expert tracking framework.

   We model the multi-expert framework using the discrete graph optimization method, where the graph node is represented by the target state hypothesis estimated by the expert. The proposed discrete graph framework is able to select the most reliable expert online by implicitly analyzing the trajectories given by the multi-expert. After defined the unary and binary graph score, the expert selection is conducted by evaluating graph path of the highest score. With the high efficient solver of dynamic programming, the proposed method can analyze the trajectories and reliabilities of the multi-expert by only computing their scores in the current frame. Three base trackers are integrated into the proposed multi-expert framework to validate its generalization. The experiments on OTB-50, OTB-100 and VOT2015 datasets demonstrate the proposed method is very effective.

At the end of this chapter, we introduce the organization for the rest of the thesis.

In this chapter, research background and literature review are introduced, and the tracking drift problem is analyzed in detail. At the last part of this chapter, we summarize the contributions of the thesis.

The rest of the thesis is organized as shown in Fig. 1.3. In accordance to the raised three issues in Sec. 1.4, Chapter 2 is the Weber's Law Shape Descriptor for object tracking. Chapter 3 is the time varying metric learning for object tracking. Chapter 4 is the tracker snapshot based

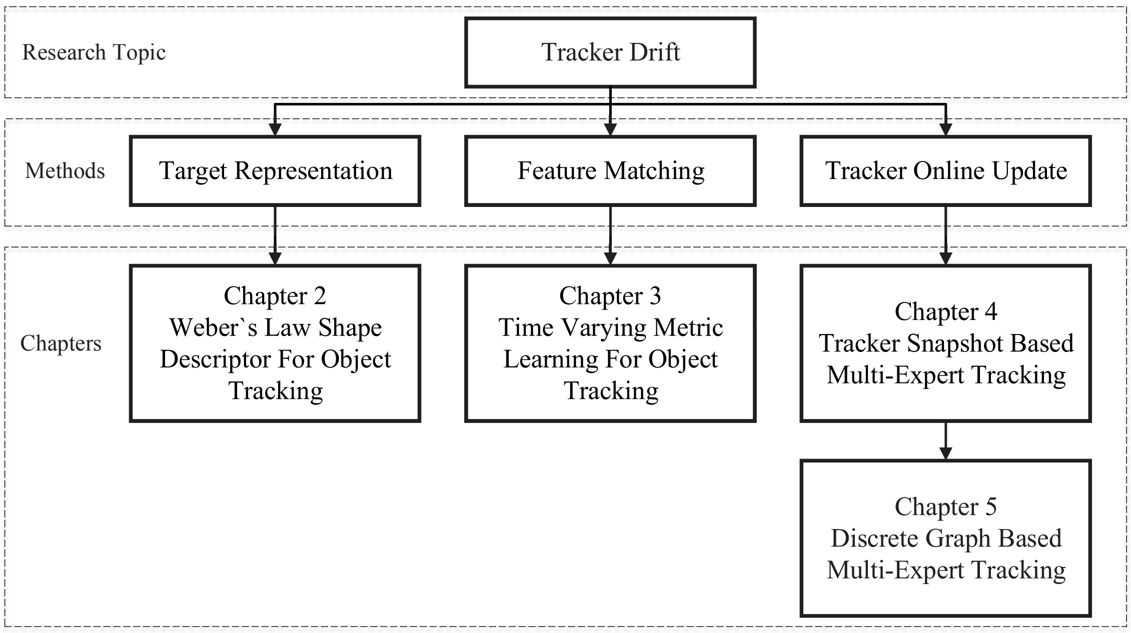

Research Topic
Tracker Drift
Methods
Target Representation
Feature Matching
Tracker Online Update
Chapters
Chapter 2 Weber`s Law Shape Descriptor For Object Tracking
Chapter 3 Time Varying Metric Learning For Object Tracking
Chapter 4 Tracker Snapshot Based Multi-Expert Tracking
Chapter 5 Discrete Graph Based Multi-Expert Tracking

Fig. 1.3 Thesis organization.

multi-expert tracking, and Chapter 5 extends the work of Chapter 4 to propose the discrete graph based multi-expert tracking.

## 1.5 Publications Related to the Thesis

1. **Jiatong Li**, Baojun Zhao, Linbo Tang, Chenwei Deng, Lu Han, Jinghui Wu. “WLSD: A Perceptual Stimulus Model Based Shape Descriptor.” KSII Transactions on Internet and Information Systems: TIIS, vol. 8, no. 12, pp. 4513- 4532.
2. **Jiatong Li**, Baojun Zhao, Chenwei Deng, Richard Xu. “Time Varying Metric Learning for Visual Tracking.” Pattern Recognition Letters, vol. 80, pp. 157-164, 2016.
3. **Jiatong Li**, Zhibin Hong, Baojun Zhao. “Robust Visual Tracking by Exploiting the Historical Tracker Snapshots.” Proc. of IEEE International Conference on Computer Vision Workshops, pp. 41-49, 2015.
4. **Jiatong Li**, Chenwei Deng, Richard Xu, Dacheng Tao, Baojun Zhao. “Robust Object Tracking With Discrete Graph Based Multiple Experts.” IEEE Transactions on Image Processing, 2016. (accepted)

# Chapter 2

# Weber's Law Shape Descriptor for Infrared Object Tracking

## 2.1 Introduction

Shape descriptor is one of the important features in infrared object tracking, especially in the tracking-by-detection methods, efficient shape feature is capable of improving the detection accuracy. However, how to establish low complexity and effective shape feature extraction method is still a challenging task. A robust shape descriptor is required to be invariant to translation, rotation and scale as well as insensitive to noise, tolerance to distortion and even occlusion. In addition, object tracking also requires the shape extraction method with low computation load.

Many shape representation and analysis methods have been proposed in the past decades. They can be generally classified into two categories: contour-based methods and region-based methods. Among the two categories, contour-based methods are in the majority [56, 57, 58]. Contour-based methods are more popular than region-based ones since they are relatively more discriminative. Contour-based shape analysis can further be divided into global description and local description. Usually, local descriptors are more discriminative than global ones. However, most local description methods aim to find correspondence of point sets of contours, consequently sacrificing computation efficiency, which is not be suitable for object tracking.

Multi-scale methods are an important research branch of shape analysis and have shown high discriminative ability [56, 59, 60, 61, 62, 63]. However, most of the existing multi-scale methods still use the scale normalization to get the scale invariance, and circular shift matching to achieve the rotation invariance, all of which add to high computation load. In addition, the general multi-scale shape representation itself is another step that is time-consuming. For example, CSS [56, 59] achieves the multi-scale representation by smoothing the shape contour using Gaussian kernels. The same representation method is also adopted by [63].

In order to develop an high efficient shape descriptor in object tracking, this chapter proposes a new contour-based shape descriptor, called Weber's Law Shape Descriptor (WLSD). WLSD is a global shape descriptor which is as discriminative as the state-of-the-art local descriptors while with very low computation complexity. In addition, WLSD is intrinsically extended to multi-scale without extra computation load, and it represents a shape by only 24 dimension vector each scale, which leads to efficient shape matching without scale normalization or circular shift.

WLSD aims to describe the salient variation of a shape contour that stimulate human perception. According to Weber's Law, under the fact that human perception of a pattern depends not only on the stimulus difference but also on the ratio of the relative stimulus change to the original stimulus intensity, we first design Contour Angular Feature (CAF) to model the so-called stimulus intensity, and then establish WLSD according to the Weber's Law Equation. WLSD is then extended to multi-scale to enhance its description capacity. We also adopt the feature selection method to select the effective scales of WLSD.

Since the mechanism of the infrared and visible imaging are different, the infrared target texture is not distinctive enough. Therefore, most common feature extraction methods designed for visible objects may not suitable for the infrared ones, which needs to develop specific feature for the infrared targets. On the other hand, target trajectory is one of the most important clues in infrared object tracking. It can also be observed that multi-feature integration is capable of improving the tracking accuracy. This chapter adopts the WLSD shape descriptor in infrared object tracking, and proposes multi-feature integration tracking method including target shape, area and moving trajectory. The experiments are first conducted in the two general shape datasets MPEG-7 and Tari, and the proposed WLSD method is compared with many state-of-the-art shape descriptors to demonstrate its discriminative ability. Then we validate the proposed multi-feature integration tracking method on the infrared target videos.

## 2.2 Related Work for Shape Descriptors

As discussed in the above section, shape feature extraction methods can be divided into contour based methods and region based methods.

There are many existing contour-based methods. One of the classic methods is Curvature Scale Space (CSS) [56]. The CSS uses the zero-crossings of the contour curvature to divide the whole shape contour into convex/concave arcs. CSS smoothens the contour by Gaussian kernels of increasing scales, and finally the whole contour will be convex with pairs of the curvature zero-crossing points evolving together. The whole evolving process of the curvature zero-crossings can be illustrated by the CSS Image, whose similarity is used for shape matching. Another well-known shape descriptor is Shape Context (SC) [57]. At each reference point, SC uses the spatial location of remaining contour points to form the 2D histogram, and thus finds the correspondence by the histogram sets of two shapes. In order to tackle shape articulation, Inner Distance Shape Context (IDSC) [58] extends the SC, and proposes to replace the traditional Euclidean distance by inner distance, which is the shortest path between landmark points within the shape silhouette, and dynamic programming is used to preserve the ordering of the contour points. Unlike SC and IDSC which utilize local histogram matching methods, Contour Points Distribution Histogram (CPDH) [64] adopts one histogram to describe the distribution of the whole contour points of a shape under polar coordinate, and the shape similarity is obtained by Earth Mover's Distance (EMD). Xu et al. [65] propose a method called Contour Flexibility, which describes the shape contour points by their deformable potential and also uses dynamic programming for shape matching. Besides designing discriminative shape descriptors, some work focus on shape matching. One of them is Hierarchical Procrustes Matching (HPM) [60]. HPM is proposed to compare shapes hierarchically by using the longer segment matching result to predict the corresponding shorter segment. In [61], Shape tree (ST) describes the contour by hierarchically dividing the shape contour into short sub-curves, and adopts the elastic matching method for shape comparison. The ST representation has good tolerance to shape distortion, and random shape deformations can be obtained by adding noise to the nodes in a shape tree without influencing to perceptually identify the shape category.

The other major category of the shape descriptor is region-based methods. For example, the Zernike moments (ZM) [66] is selected as the MPEG-7 standard region-based shape descriptor, but it is relatively time consuming and has limited tolerance to shape distortion. Another moment-based shape descriptor is Moment Invariant (MI) [67]. Although MI performs not as well as ZM, it only has 7 feature vectors, with low storage and high speed retrieval response, so it is also a widely used shape analysis method. Recently, Support Vector Shape (SVS) [68] is proposed which uses the decision function trained by Support Vector Machine (SVM) to describe the shape. With the Radial Basis Function (RBF), SVS maps points within the shape to high dimension, and has good performance against severe noise.

Recent years, new trends in shape analysis have emerged. For instance, some work focus on the post-processing after shape matching [69, 70]. In [69], Bai et al. propose to construct the graph model using the similarity (or distance) of shapes pairs, and adopt the graph transduction to learn the graph structure implicitly. The context-based shape retrieval methods like graph transduction can well handle the situation in which intra-class distances are larger than inter-class ones and is capable of improving the retrieval rate based on available shape measures.

In addition, some researchers propose the heuristic-based auxiliary shape descriptors aim at describing some specific kinds of shapes. For instance, in [71], two perceptually motivated strategies are proposed. The first handles shapes with base structure and "strand" structures, and the second handles symmetry shapes. The two strategies can be integrated into existing shape matching methods to improve the retrieval or classification performance. In this chapter, we focus more on shape representation and shape matching. We do not focus on the above mentioned post-processing methods since they can be used in all available shape measures.

## 2.3 Weber's Law Shape Descriptor

In this section, we first review Weber's Law. We then propose the Contour Angle Feature (CAF) and use it to construct WLSD, and demonstrate the importance of the multi-scale property of WLSD. Finally, we introduce the WLSD scale selection scheme.

### 2.3.1 Weber's Law

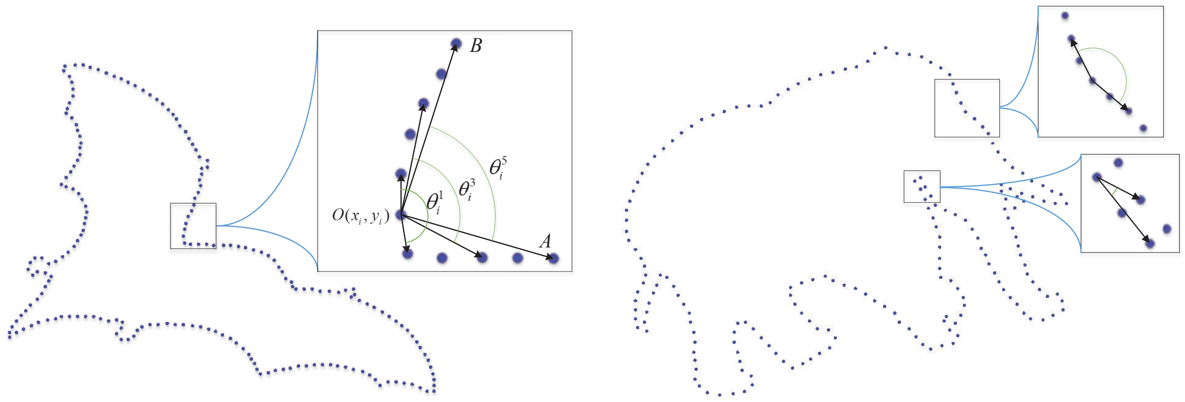
Weber's Law is a psychological rule. It demonstrates that the change in stimulus intensity varies by the original stimulus intensity, and the ratio of the just noticeable difference (JND) of the perception to the original stimulus is a constant [72, 73]. The relationship can be expressed as:

$$\frac{\Delta I}{I} = k, \tag{2.1}$$

where $\Delta I$ denotes the JND, and $I$ represents the original stimulus intensity. $k$ is the constant corresponding to the certain perception. The fraction is known as the Weber fraction or Weber proportion. For example, an experiment by Weber in 1840 shows that people can feel the increase or decrease of the additional 1 gram if they hold a 52 grams weight object, but can only feel 2 gram if they hold 104 grams, which demonstrates the JND differs according to the original stimulus intensity.

### 2.3.2 The Original Stimulus Feature - Contour Angle Feature

Weber's Law motivate us to build a perceptual stimulus model. The first task is to construct a shape descriptor as the original stimulus intensity. Many shape descriptors have been proposed, such as the distance between the shape centroid and the contour [74], the contour curvature, signature, Fourier Descriptor (FD), which are either difficult to model Weber's Law or sensitive to scale or rotation variation, so we design a new shape descriptor called Contour Angular Feature (CAF).



(a) $CAF_s$ examples (a) $CAF_s$ varies from 0 (bottom image) to $\pi$ (upper image)

Fig. 2.1 Illustration for $CAF_s$

We first introduce the single scale CAF, and then present the general definition of the multi-scale CAF.

As illustrated in Fig. 2.1(a), given a closed contour point $O$ with its nearest right and left points $A$ and $B$, then $\angle AOB$ is called the CAF of point $O$. Note $\angle AOB$ has two directions, inward or outward relative to the closed contour, and the sum of two angles are $2\pi$. In this chapter, we suppose the CAF always be outward. CAF can be extended to the multi-scale naturally, let $CAF_s$ denotes the $s$ scale of CAF, then angle $\angle AOB$ is $CAF_1$, and $CAF_2$ is the angle formed by contour point $O$ with its second nearest right and left points, etc. The general definition of $CAF_s$ can be defined.

**CAF Definition:** Given a closed contour $C$ with $n$ samples ordered clockwise, whose coordinates are denoted as $(x_i, y_i)$, $i = 1, 2, \ldots, n$, then the $CAF_s$ of $O(x_i, y_i)$ is defined as:

$$\theta_i^s = \begin{cases} \arccos\left(\dfrac{\overrightarrow{OA_s} \cdot \overrightarrow{OB_s}}{\left|\overrightarrow{OA_s}\right| \left|\overrightarrow{OB_s}\right|}\right), & \overrightarrow{OA_s} \times \overrightarrow{OB_s} > 0 \\ 2\pi - \arccos\left(\dfrac{\overrightarrow{OA_s} \cdot \overrightarrow{OB_s}}{\left|\overrightarrow{OA_s}\right| \left|\overrightarrow{OB_s}\right|}\right), & \overrightarrow{OA_s} \times \overrightarrow{OB_s} < 0, \end{cases} \tag{2.2}$$

where $s = 1, 2, \ldots, \lfloor (n-1)/2 \rfloor$ ($\lfloor \quad \rfloor$ denotes the floor function), and $\times$ represents the cross product, $\overrightarrow{OA_s} = \left(x_{\bmod (i+s,n)} - x_i, y_{\bmod (i+s,n)} - y_i, 0\right)$, $\overrightarrow{OB_s} = \left(x_{\bmod (i-s,n)} - x_i, y_{\bmod (i-s,n)} - y_i, 0\right)$.

In the **CAF Definition**, the multi-scale $CAF_s$ can be obtained with $s$ varying from 1 to $\lfloor (n-1)/2 \rfloor$. $A_s$ and $B_s$ are the right and left points relative to $O$ respectively, and both are $s$ points away from $O$. Consequently, each sample point has the maximum of $\lfloor (n-1)/2 \rfloor$ scale angles. Examples for $CAF_1$, $CAF_3$ and $CAF_s$ of point $O(x_i, y_i)$ are shown in Fig. 2.1(a). The reason why extending $\overrightarrow{OA_s}$ and $\overrightarrow{OB_s}$ to three dimensional space is to highlight the direction property of the vector cross product (for instance, $\overrightarrow{OA_s} \times \overrightarrow{OB_s} > 0$ indicates that $B \to O \to A$ is clockwise), therefore, the definition guarantees angle $\theta_i^s$ always directing outwards relative to the interior of a closed contour. Analogously, the $CAF_s$ of the opposite direction can be defined, which is equivalent to **CAF Definition**. $\theta_i^s$ varies from 0 to $2\pi$, indicating the local contour of the shape changing perceptually from inwards strand structure to outwards one (as illustrated by Fig. 2.1(b)), so $CAF_s$ is capable of describing all the variations of a shape contour. Notice CAF feature is invariant to scale and rotation.

### 2.3.3 Weber's Law Shape Descriptor

Motivated by Weber's Law, we use the difference of CAF between the current point and its neighbors to model the difference of the stimulus intensity. At the same time, the saliency variation that stimulates human perception is modeled by the proportion of the CAF difference to the CAF value of the current point. We then adopt the arctangent function to operate on the proportion [75], which can suppress the noise. Let $\theta_i$ be $CAF_1$ of the contour point $O(x_i, y_i)$, then the saliency of point $O(x_i, y_i)$ can be described as:

$$f(\theta_i) = \arctan\left(\frac{\theta_{\bmod\ (i+1,n)} - \theta_i}{\theta_i} + \frac{\theta_{\bmod\ (i-1,n)} - \theta_i}{\theta_i}\right), \tag{2.3}$$

where $\theta_i$ denotes the $CAF_1$ of the current point, $\theta_{\bmod\ (i+1,n)}$ and $\theta_{\bmod\ (i-1,n)}$ represent the right and left neighbor points of the current point respectively. We call this feature Weber's Law Shape Descriptor (WLSD), and $f(\theta_i)$ the WLSD value of point $O(x_i, y_i)$.

The idea of the WLSD is partially motivated by Weber's Law Descriptor (WLD) proposed in [75]. WLD is an image descriptor that takes the gray intensity as the original stimulus intensity. In WLD, the difference of the stimulus is the intensity differences of a current pixel and its neighbors. While in WLSD, CAF is viewed as the stimulus intensity, and the difference of the stimulus is the CAF difference of a current contour point and its neighbors. However, WLSD extracts feature of the shape contour, which is less informative than that of gray image, and the CAF of a contour varies relatively more smoothly than the gray intensity. Therefore, it is necessary to explore the multi-scale representation of WLSD.

If we only use single neighbor based WLSD shown in Eq. 2.3, the points of large WLSD value (salient points) will only occupy a small proportion of the whole contour points. Fig. 2.2 maps the single scale WLSD value of the contour points linearly to the colormap bar so that the description capacity of WLSD can be observed intuitively (the minimum value maps to leftmost deep blue, maximum to the rightmost deep red). It can be seen that WLSD presents a sparse

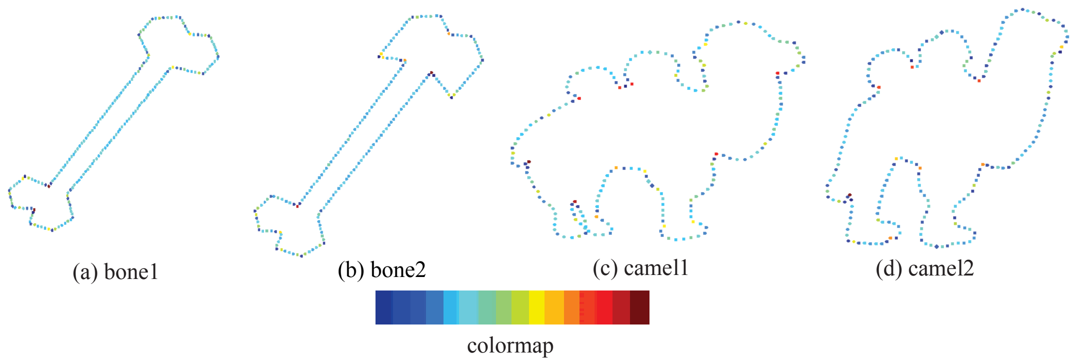


Fig. 2.2 $WLSD_{1\to 1}$ (single scale WLSD) value linearly mapped to the colormap bar

distribution, with most of the WLSD value focus on the middle of the colormap bar (nearly 0). In addition, the visualization of single neighbor based WLSD presents an appearance of abrupt variations because of noise and the continuity variation of the shape contour, which is unstable and less discriminative. In order to obtain a relatively continuous and robust description of the contour in accordance to human perception, we extend WLSD to multi-scale.

The scale of WLSD has a close relationship with the scale of CAF, let $WLSD_w$ represents WLSD of scale $w$. Given a contour sampled with $n$ points and its corresponding $CAF_s$, let $w = 1$, then $WLSD_1$ of point $O(x_i, y_i)$ is:

$$f(\theta_i^s) = \arctan\left(\frac{\theta^s_{\text{mod }(i+s,n)} - \theta_i^s}{\theta_i^s} + \frac{\theta^s_{\text{mod }(i-s,n)} - \theta_i^s}{\theta_i^s}\right), \tag{2.4}$$

where $\theta_i^s$ denotes $CAF_s$ defined as in Eq. 2.2. Consequently, each $WLSD_w$ has a corresponding scale of $CAF_s$. At this point of view, the scale of WLSD is also determined by the scale of $CAF_s$ implicitly. We use $WLSD_{s\to w}$ to represent the $WLSD_w$ extracted from $CAF_s$, then the above definition in Eq. 2.4 can be denoted as $WLSD_{s\to 1}$. Then we introduce the general definition of $WLSD_{s\to w}$:

**WLSD Definition:** Given a closed contour $C$ with $n$ samples ordered clockwise, whose coordinates are denoted as $(x_i, y_i)$, $i = 1, 2, \ldots, n$, then the $WLSD_{s\to w}$ of $O(x_i, y_i)$ is defined as:

$$f(\theta_i^s) = \arctan\left(\sum_{k=1}^{w} \frac{\theta^s_{\text{mod }(i+k\times s,n)} - \theta_i^s}{\theta_i^s} + \sum_{k=1}^{w} \frac{\theta^s_{\text{mod }(i-k\times s,n)} - \theta_i^s}{\theta_i^s}\right), \tag{2.5}$$

where $\theta_i^s$ denotes $CAF_s$ of $O(x_i, y_i)$, and $w = 1, \ldots, \lfloor\lfloor(n-1)/2\rfloor/s\rfloor$ ($\lfloor\quad\rfloor$ denotes the floor function).

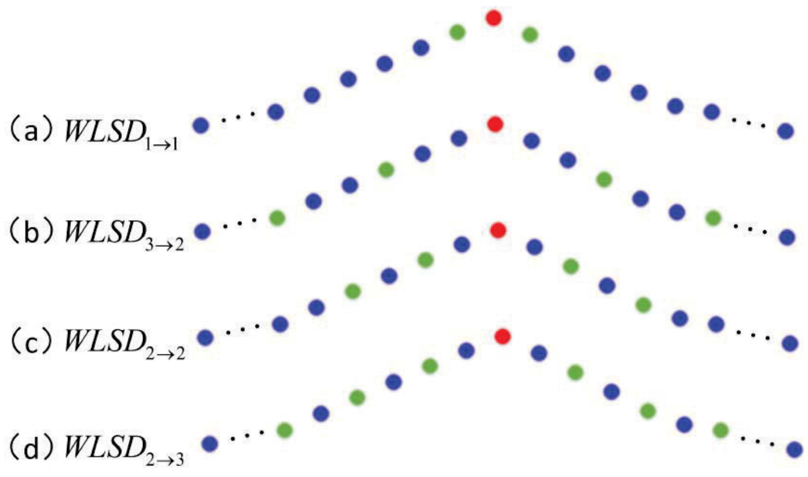


Fig. 2.3 Illustration for $WLSD_{s\to w}$

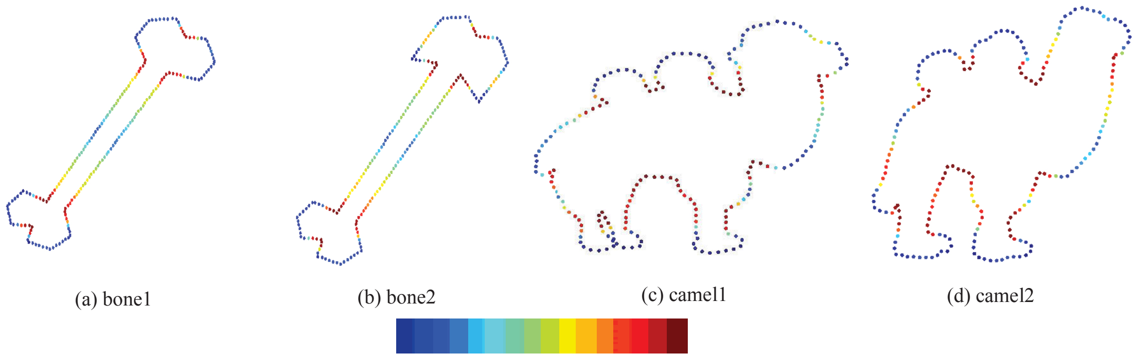


Fig. 2.4 $WLSD_{5\to5}$ visualization

Therefore, given the $CAF_S$ of a contour, the stimulus intensity difference of the current scale $w$ is the difference between the current point's $CAF_s$ and the $CAF_s$ of its right and left $w$ points. Fig. 2.3 illustrates some examples of $WLSD_{s\to w}$. The current point is colored red, and the right and left $w$ number of points corresponding to different scale are colored green.

Fig. 2.4 illustrates that multi-scale WLSD can describe the contour saliency well and has the desired continuous saliency variation description property. It can be seen that the multi-scale WLSD has relatively uniform distribution within the WLSD value range and can describe the contour saliency continuously. Fig. 2.4 also indicates that WLSD presents similar visualization for intra-class shapes.

### 2.3.4 WLSD Histogram and WLSD Scale Selection

In order to get the WLSD feature that is tolerant to the shape distortion, we uniformly divide the $WLSD_{s\rightarrow w}$ value range, and use one dimensional histogram as the global shape feature. Given two histograms $H_1$ and $H_2$, we adopt the commonly used $\chi^2$ distance to compare features:

$$d(H_1, H_2) = \frac{1}{2}\sum_{k=1}^{K}\frac{[h_{1k}-h_{2k}]^2}{h_{1k}+h_{2k}}, \tag{2.6}$$

where $K$ represents the number of histogram bins. $h_{1k}$ and $h_{2k}$ denote the number of points falling into the $k$th bin of $H_1$ and $H_2$ respectively. In this paper we set $K = 24$.

Different scale of WLSD contains different discriminative information, for instance, $WLSD_{3\rightarrow 3}$ is able to capture the local information of the contour, while $WLSD_{16\rightarrow 5}$ tends to describe the contour in a global view. As mentioned above, each $WLSD_w$ has a corresponding scale of $CAF_s$. Given a contour of $n$ points, then the whole number of scales of CAF is $\lfloor (n-1)/2 \rfloor$, and each scale of the CAF corresponds to $\lfloor \lfloor (n-1)/2 \rfloor / s \rfloor$ scales of WLSD. Consequently, given a $n$ points contour, we can get the whole number of scales of WLSD, which is denoted by $S_{WLSD}$:

$$S_{WLSD} = \sum_{s=1}^{\lfloor (n-1)/2 \rfloor} \lfloor \lfloor (n-1)/2 \rfloor / s \rfloor. \tag{2.7}$$

We aim to select the discriminative scales of WLSD among the whole number of scales shown in Eq. 2.7, and combine the selected scales to be the final effective features. Motivated by the feature selection algorithms, we adopt the Sequential Forward Selection (SFS) [76] to select effective WLSD scales. First, we define the feature evaluation criterion according to specific application, and then select the scale in the candidate set iteratively that can improve the defined criterion until the improvement stops. The scales selected by SFS may not be optimal, but very efficient, which will be verified in the experiments.

One of the problem is that the feature dimension will expand as the iterative times increase, leading to the increasing computation load of shape matching. To overcome the above problem, we operate directly on the distance matrix instead of calculating the combined features. Suppose there are $M$ training shapes, then the distance matrix is $\mathbf{D}_{M\times M}$, whose elements are the distance between each pair of shapes. Suppose each contour is uniformly sampled by 200 points, then the size of the initial candidate set $S_{WLSD}$ is 473 according to Eq. 2.7, which can be denotes by:

$$\begin{aligned}
&\mathbf{D}_{1\to 1}, \mathbf{D}_{1\to 2}, \ldots, \ldots, \mathbf{D}_{1\to 99}, \\
&\mathbf{D}_{2\to 1}, \mathbf{D}_{2\to 2}, \ldots, \mathbf{D}_{2\to 49}, \\
&\ldots \\
&\mathbf{D}_{99\to 1}.
\end{aligned} \tag{2.8}$$

The subscript of $\mathbf{D}$ is in accordance to that of $WLSD_{s\to w}$, and it can be simplified by $\{\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_l, \ldots, \mathbf{D}_{S_{WLSD}}\}$. The whole WLSD scale selection steps are shown in the following algorithm chart. The evaluation criterion $J(g)$ is based on specific application. Here, we choose this criterion as bull's eye score which will be detailed in Sec. 2.5.

**Algorithm 1:** WLSD Scale Selection Algorithm

```
Initialization: define the evaluation criterion $J(g)$;
                evaluate the candidate distance matrix set $\{\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_l, \ldots, \mathbf{D}_{S_{WLSD}}\}$;
                initialize the selected distance matrix set $\mathbf{W}_0 = \varnothing$, selected scale $\mathbf{L} = \varnothing$,
                and the iterative counter $k = 0$;
while 1 do
    Select the distance matrix $\mathbf{D}_{l_k} = \arg\max(J(W_k + \mathbf{D}_{l_k}))$ in the candidate set;
    if $J(W_k + \mathbf{D}_{l_k}) < J(W_k)$ or $k > S_{WLSD}$ then
        break;
    else
        Add $l_k$ to the selected scale $\mathbf{L}$;
        Update the selected set $\mathbf{W}_{k+1} = \mathbf{W}_k + \mathbf{D}_{l_k}$;
        Remove $\mathbf{D}_{l_k}$ from the candidate set;
        k = k + 1;
Get the selected scale $\mathbf{L}$ of $WLSD_{s\to w}$.
```

After the above training procedures, the selected scales are recorded in $\mathbf{L}$. In specific application, for example, shape retrieval, we first train the samples to acquire the effective WLSD scales, and then compare shapes according to their combination distances of the selected WLSD scales.

## 2.4 WLSD for Object Tracking

### 2.4.1 Infrared Target Representation

Feature extraction methods are commonly used for the target appearance representation in optical image, including the keypoint feature [77], Histogram of Oriented Gradients (HOG) [78] feature and the state-of-the-art deep feature [79]. However, the above appearance feature extraction methods usually don't work very well in the infrared image object representation, since the infrared targets lack rich texture information. Therefore, this chapter propose to use the multiple features of target shape, area and trajectory for infrared target representation.

The gray level of infrared image lies on the thermal radiation intensity of the infrared target because of the passive imaging property. The existed thermal difference makes it relatively easier to distinguish the infrared target from the background. As a result, we can take advantage of this to segment the target from the background. In addition, there usually exists local maxima on the infrared target silhouette, which can be used to search for the target candidates. After that, the target features of shape, area and trajectory can be further used to exclude the false alarms.

### 2.4.2 Multi-feature Integration Tracking

#### Tracking Initialization

In order to get the precise location, shape and area of the target, we adopt the target segmentation scheme to refine the object silhouette given the initial target bounding box. Traditional global based segment methods like Otsu's method and adaptive global thresholding method will cause multiple disconnected components. Region growing segmentation [80] generates one component only, which is more suitable for infrared object segmentation.

Region growing segmentation searches neighboring pixels of seed points iteratively and determines whether the pixel neighbors should be added to the region. The region growing criteria is:

$$|I(x,y) - I(x_s,y_s)| < T_g, \tag{2.9}$$

where $I(x,y)$ and $I(x_s,y_s)$ denote the comparing pixels and the gray scale of the seed point respectively, and $T_g$ represents the similarity threshold. When region growing is finished, we can get the initial target area, center and shape silhouette, then WLSD feature is extracted to describe the target shape contour.

#### Target Searching

We adopt the tracking-by-detection scheme to search the target candidates, and use the multi-feature integration methods to exclude false alarms. The whole target searching flowchart is shown in Fig. 2.5. The searching steps are detailed as follows:

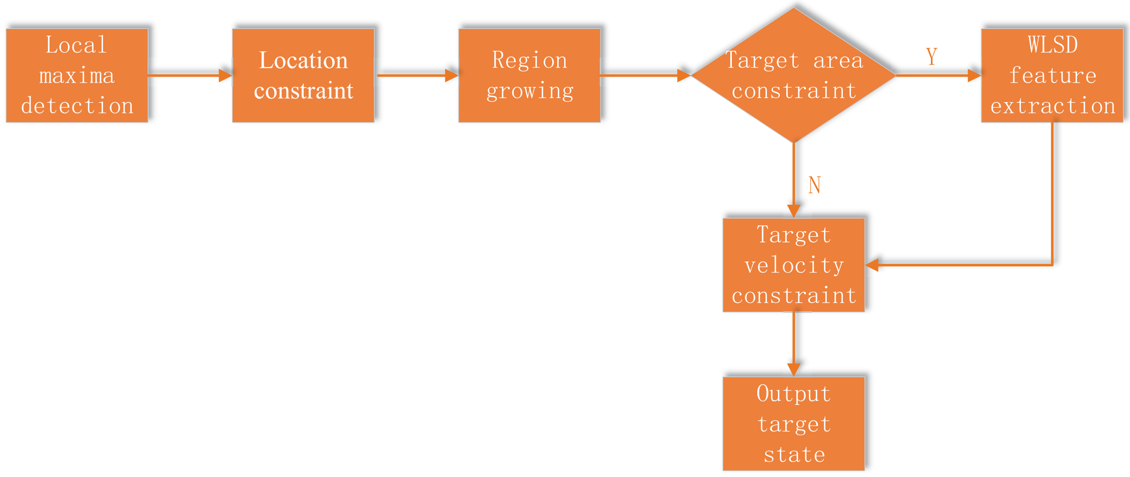


Fig. 2.5 Target Searching flowchart

1. Local maxima detection
   Given one point $P(x,y)$ of gray scale $I(x,y)$ within the image, if

$$I(x,y) \geq \max(I(x-\lfloor n/2 \rfloor : x+\lfloor n/2 \rfloor, y-\lfloor n/2 \rfloor : y+\lfloor n/2 \rfloor)), \tag{2.10}$$

   then $P(x,y)$ is the local maximum within the $n \times n$ neighbor pixels.

   It is notable that there often exist the so-called parallel local maxima which means that pixels within a small region have the same or near the same maximum gray scale. The existence of parallel local maxima will increase the computation load. Therefore, We adopt the region growing method to cluster parallel local maxima. If the distance between the seed point and the local maxima is less than a pre-defined threshold $T_m$, then the two points are in the same cluster. After region growing, the location of the same cluster is represented by the mean coordinates of all the local maxima in the cluster.

2. Location constraint
   The distance of the target in adjacent frames can not be very large assuming the target velocity is limited. Therefore the location constraint can exclude many false alarms. The local maxima whose distance from the target location in the previous frame less than $T_l$ are regarded as the target candidates, then they are conveyed to the next step to make further decision.

3. Region growing, target area constraint
   In these two steps, the target candidates are first segmented using region growing introduced at the beginning of this Section, and are refined under the area constraint $|A_i - A_{t-1}| \leq T_a$,

where $A_i$ is the area of target candidate, and $A_{t-1}$ is the target area of the last frame, $T_a$ is the area threshold.

4. WLSD feature extraction, target velocity constraint and the target state output
   It is not suitable to extract the shape feature when the target area is too small. Therefore, if the target area doesn't satisfy the constraint, we directly choose the candidate of the closest velocity as the target location, otherwise, WLSD feature of the target shape is extracted.

In summary, after the initialization for target as described in the "Tracking Initialization" part of Sec. 2.4.2, we track the target as the searching steps shown in the "Target Searching" part of this Section.

## 2.5 Experimental Results

In this section, the proposed WLSD feature is first tested in MPEG-7 and Tari [81] shape database for shape retrieval, and is compared with many state-of-art methods. Then the proposed multi-feature integration tracking method is validated in the infrared tracking videos.

In the MPEG-7 dataset, we further analyze WLSD in respect of the computation complexity, the robustness of the scale selection approach and the effectiveness of WLSD compared with CAF.

In the following experiments, the retrieval rate is measured by the so-called bull's eye score. Let $t$ be the number of shapes of the same class. Every shape in the database is compared to all other shapes, and the number of shapes from the same class among the $2t$ most similar shapes is reported. The bull's eye retrieval rate is the ratio of the total number of shapes from the same class to the highest possible number (in MPEG-7 it's $20 \times 1400$). We adopt the five-fold cross validation to demonstrate the performance of WLSD. The dataset is divided into five folds, one split for validation and the others for training. Note that each validation fold is tested on the whole dataset using the scales obtained from the other four training splits. Consequently, the combination of the five validation folds test will cover the retrieval results of all the dataset samples.

Each contour is uniformly sampled by 200 points. According to Eq. 2.7, the number of the initial candidate set $S_{WLSD}$ is 473. The experimental platform is on Matlab 7.11.0, with a PC of Intel(R) Core(TM) 2 Duo CPU, 2.53 GHz.

### 2.5.1 Evaluation on MPEG-7 Dataset

MPEG-7 (short for MPEG-7 CE-Shape-1 Part B) dataset is widely used in shape retrieval, it consists of total 1400 shapes, having 70 shape classes, and each class has 20 instances.

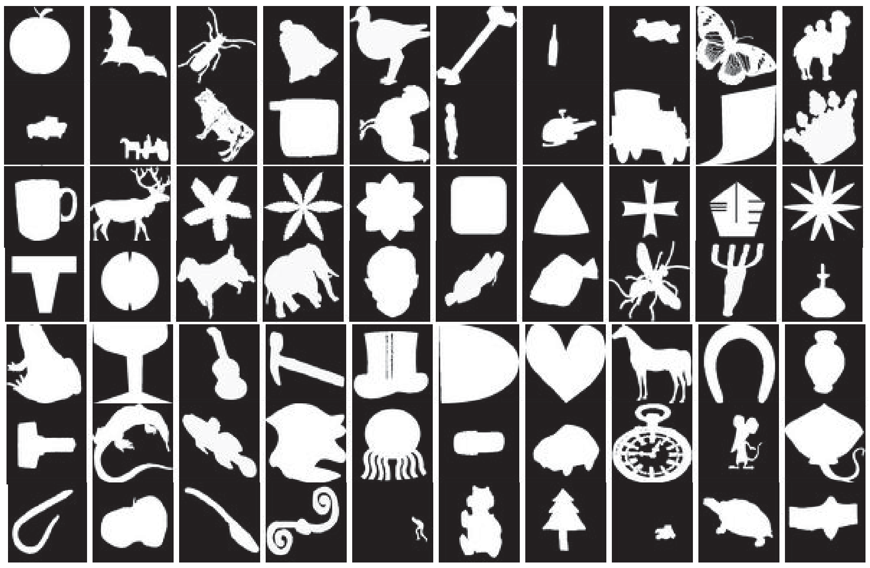


Fig. 2.6 MPEG-7 examples

The database is challenging due to the presence of shape distortions and examples that are perceptually dissimilar from other inner-class instance and highly similar inter-class ones. The first instance of each class is shown in Fig. 2.6:

**Compare with the Existing Methods**

In this section, WLSD is compared with the recent 11 shape analysis methods. From Table. 2.1, it can be seen that WLSD performs the best among all the compared methods, including multi-scale methods, like Hierarchical Procrustes [60], Multi-scale Representation [62] and Polygonal Multi-resolution [63]. Moreover, WLSD is much more efficient than all the methods listed on the table. In real application, after the scale selection and the feature extraction for the database having been done off-line, the computation load of shape retrieval using WLSD is only cost by feature extraction of the query shape, and the 24-dimensional vector distance computation for selected scales. Under the above experimental environment, the time cost by WLSD pairwise shape matching is only 2.2ms with 200 contour points.

Among the methods whose scores are above 85%, Hierarchical Procrustes reports 300ms taken by matching two shapes [60]. The computation time for pair-wise shape similarity in Symbolic takes 76.5ms on average, with 100 contour points [82]. Shape L'Ane Rouge estimates the density takes on average 2 to 3 minutes per shape, which is also much more time consuming than WLSD [83]. Represented by 100 points, IDSC reports 0.31s on a 2.8G PC implemented by optimized Matlab code [58]. There are two reasons for time efficiency of WLSD, one is that

Table 2.1 Overall performance comparison

| **Compared methods** | **Score (%)** |
|---|---|
| Hierarchical Procrustes [60] | 86.35 |
| Symbolic [82] | 85.92 |
| IDSC [58] | 85.40 |
| Multi-scale Representation [62] | 84.93 |
| Polygonal Multi-resolution [63] | 84.33 |
| Shape L'Ane Rouge [83] | 82.25 |
| DSW [84] | 82.13 |
| GMS [85] | 80.03 |
| CPDH [64] | 76.56 |
| SC [57] | 76.51 |
| CSS [56] | 75.44 |
| **WLSD** | 86.65 |

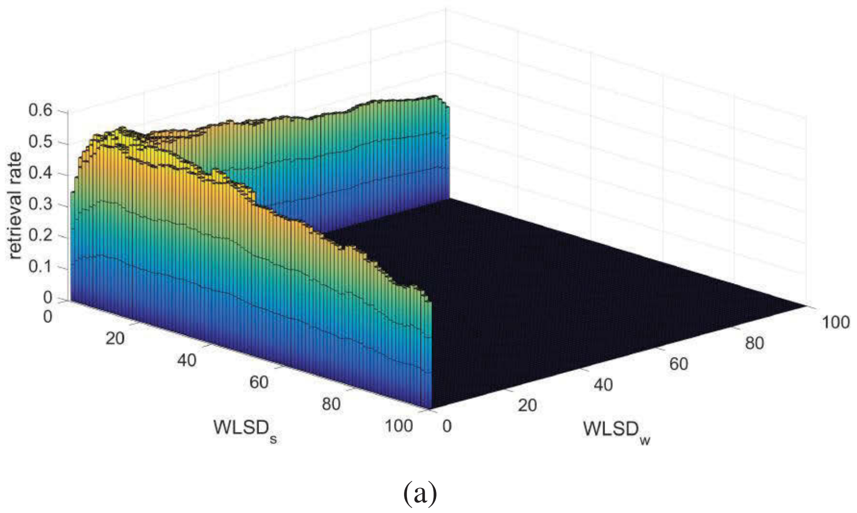
red: rank 1, green: rank 2, blue: rank 3

WLSD is a global shape descriptor which is much faster than local descriptors that depend on point-wise matching; on the other hand, unlike traditional methods such as [56] and [62] that adopt complex operations to obtain multi-scale representation, WLSD is intrinsically extended to multi-scale without extra extension load.

### Experiments for Single Scale WLSD

We first test the effectiveness of the single scale WLSD by showing its bull's eye retrieval rate of each scale in Fig. 2.2 (a). The scales mainly spread near the coordinates according to the relationship property between $WLSD_s$ and $WLSD_w$. From the figure, it can be seen that the highest score of single scale WLSD is $WLSD_{10\rightarrow5}$ with score of 58.30%, while the lowest score is only 33.16%. The results demonstrate that the performance of single scale WLSD is not satisfactory. Consequently, the multi-scale fusion is necessary. From Fig. 2.2 (a), it can also be seen that the WLSD features of high scores are mainly focused on the low scales. The score decreases gradually as the scale becomes larger.

We also want to observe the performance of $WLSD_s$ and $WLSD_w$ respectively. So in Fig. 2.2 (b) and (c), the front and the left side elevation of Fig. 2.2 (a), the illustration of $WLSD_s$ and $WLSD_w$ are drawn respectively. It illustrates that $WLSD_w$ scales of high scores mainly range from 10 to 40, and the counterparts of $WLSD_s$ range from 1 to 20. Although it seems $WLSD_s$ performs better than $WLSD_w$, it does not mean that $WLSD_w$ is not important, as it can be



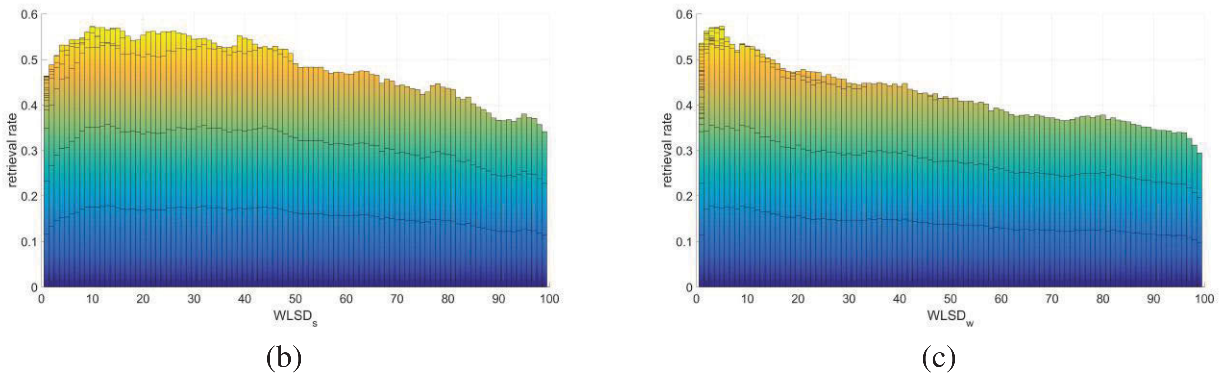


Fig. 2.7 (a) The retrieval rate of single scale WLSD; (b) The front elevation of (a); (c) The left side elevation of (a);

seen from Fig. 2.2 (a) that only after $WLSD_w$ scale starts to grow can $WLSD_{s\to w}$ reaches an increasing performance. It demonstrates that $WLSD_s$ and $WLSD_w$ make good combination, and $WLSD_w$ makes necessary supplement to $WLSD_s$.

## Experiments for Multi-Scale WLSD

1. **Multi-scale integration**. In the first experiment of this section, we will demonstrate the effectiveness of the multi-scale integration. First, the detail of the iteration performance is discussed. In Fig. 2.8 (a), the curve shows that the retrieval rate using WLSD increases with iteration times grows. It demonstrates that the power of a single scale WLSD is not very satisfactory but the score increases shapely to 73.41% just after the second iteration. The retrieval rate arrives more than 80% with less than 4 scales integration and reaches above

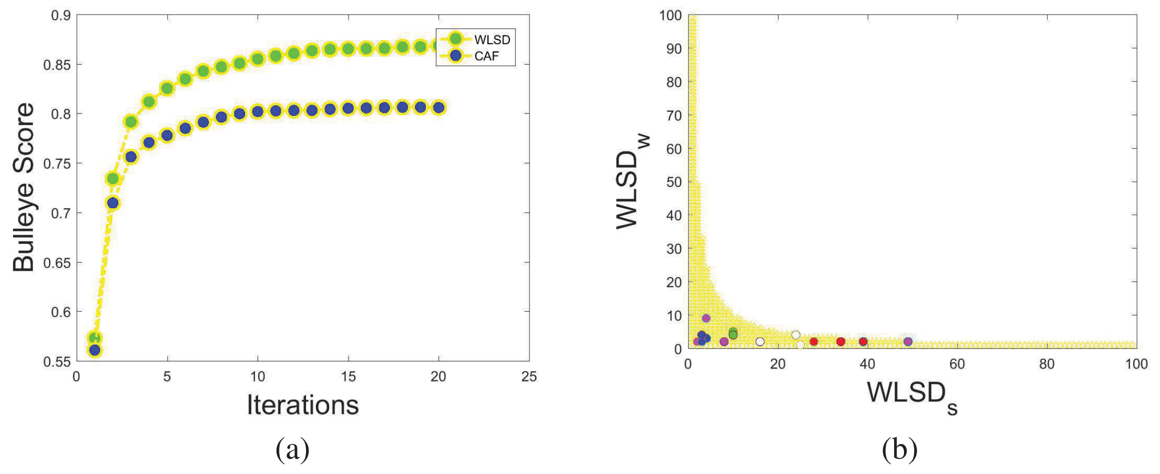


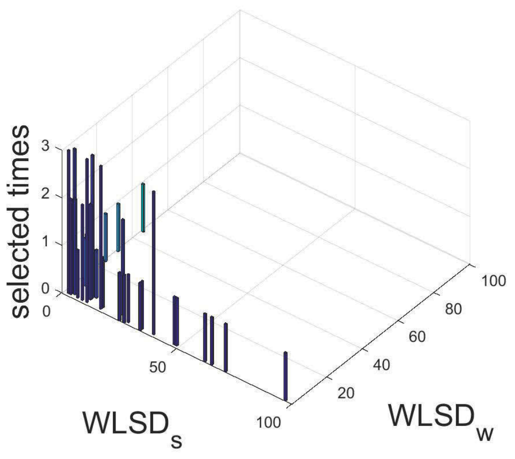
Fig. 2.8 (a) The bulleye score variation with iteration increasing; (b) The first five selected scales of each training set in MPEG-7.

85% with less than 8 scales, which shows the efficiency of SFS. Finally, after ten iterations, the performance is relatively stable.

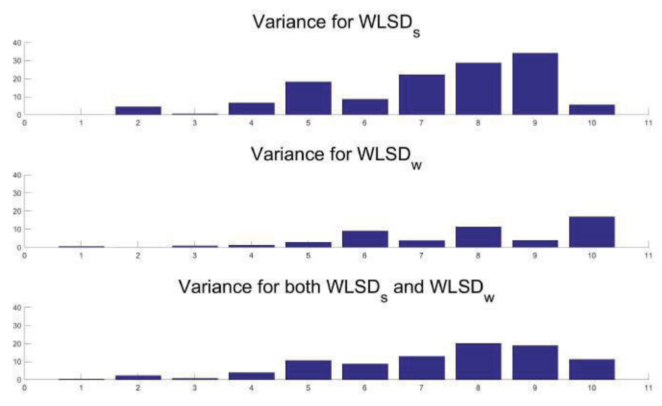
Then, we demonstrate the effectiveness of Weber's Law applied to WLSD, the results of retrieval rate using only the CAF is shown by the blue curve in Fig. 2.8 (a). The highest score obtained by CAF is 80.59%, which is much lower than WLSD. But the increasing trend of the retrieval rate is similar to WLSD. Overall, WLSD can dramatically improve the performance compared with CAF. The main reason is that WLSD can further describe the saliency variation of the shape contour whereas CAF cannot. The same increasing trend of the curves for both WLSD and CAF shown in Fig. 2.8 (a) also indicates that using SFS for WLSD scale selection is efficient and stable.

The distribution of the first five selected scales of the five-fold subsets is shown in Fig. 2.8 (b), where the selected scales are sequentially colored by green, blue, red, white and magenta. It can be seen that the variance of the selected scales becomes larger as the number of iteration increases, and the variance along $WLSD_s$ is obvious than $WLSD_w$, indicating lower scales of $WLSD_{s\to w}$ are relatively more stable.

2. **Scale selection robustness**. In the second experiment, we demonstrate the robustness of the proposed scale selection algorithm. In Fig. 2.9 (a), the accumulation of the selected scales of the five training sets are illustrated. Compared with Fig. 2.8 (b), as more scales are included, the scale of $WLSD_s$ spreads more uniformly than $WLSD_w$. It can be seen that once the high scale of $WLSD_s$ is selected, the high scale of $WLSD_w$ is not necessary to some extent, or in another way, high scales of $WLSD_w$ are not as discriminative as the lower



(a)



(b)

Fig. 2.9 (a) The first ten selected scales accumulation of the training sets; (b) The variance of the first ten selected scales.

ones. The quantitative analysis for the variation of the selected scales is also conducted. In Fig. 2.9 (b), the variance of the first ten selected scales for the training sets are illustrated. From the first two figures, it can be seen that the variance of the first several selected scales of both $WLSD_s$ and $WLSD_w$ are very stable. Although there are some fluctuations in higher scales, the whole stability is satisfactory. We conclude that the proposed scale selection algorithm is stable in the dominant discriminative scales and have acceptable variation in the complementary scales.

As shown in Fig. 2.9 (b), $WLSD_w$ is relatively more stable than $WLSD_s$. Although $WLSD_s$ performs better than $WLSD_w$, $WLSD_w$ makes more contribution in the stability of the whole $WLSD_{s\to w}$. We further draw the conclusion that $WLSD_s$ dominates the discriminability and $WLSD_w$ not only provides necessary supplement but also makes $WLSD_{s\to w}$ more stable, which demonstrates again the close relation between $WLSD_s$ and $WLSD_w$.

### 2.5.2 Evaluation on Tari Dataset

In order to demonstrate that WLSD can also handle articulated shapes, we test the proposed WLSD on the new Tari dataset. This consists of 1000 binary images with 50 shape categories, and each category has 20 images. The Tari dataset is designed with large intra-class shape deformation and many shapes are articulated (shown in Fig. 2.10).

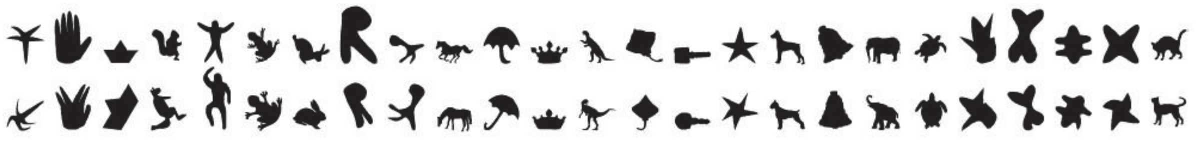


Fig. 2.10 Tari dataset examples

Table 2.2 Overall performance comparison on Tari dataset

| **Compared methods** | IDSC [58] | COP [86] | SPM [87] | DSW [84] | **WLSD** |
|---|---|---|---|---|---|
| **Scores(%)** | 95.33 | 92.18 | 91.37 | 81.60 | 93.09 |

### Overall Performance

In the overall performance for Tari dataset, we also adopt the bull's eye score. The experimental results are shown in Table. 2.2. From Table. 2.2, the score of IDSC is higher than WLSD since it is designed to handle articulated shapes and is insensitive to articulation. Overall, WLSD is comparable to the listed methods, which validates the proposed method can handle articulated shapes well.

The iteration plot and the distribution of scales are illustrated in Fig. 2.11. The curve in Fig. 2.11 (a) shows that after the second iteration, the retrieval rate increases significantly and begins approaching to the highest value. The selected scale clusters in lower scales and disperses in the higher scales. Both of the figures are in accordance with the experiments in MPEG-7 dataset, which validates the conclusion made in MPEG-7 experiment, and also shows the stability of the proposed method.

The variance test shown in Fig. 2.12 demonstrates that the variance of $WLSD_w$ is more stable than $WLSD_s$. Compared with Fig. 2.8 (b), the average variance here is smaller, since the shape samples in Tari dataset have less distortions than MPEG-7. In summary, the overall test results present the same variation as those in MPEG-7 test. It demonstrates again the robustness of the scale selection method.

### Rotation and Scale Invariance

As introduced before, both CAF and WLSD are insensitive to rotation and scale variation, we demonstrate this property in this experiment. The experiment is conducted on transformed Tari dataset by rotating the shape samples to 30°, 60°, 90° clockwise, and scaling the samples to 0.5, 1.5 and 2. The test results are shown in Table. 2.3. From the table, it can be seen that the rotated and the scaled scores are nearly the same to the original ones (the rotation 0° and scale 1), which demonstrates CAF and WLSD are robust to the rotation and scale variation.

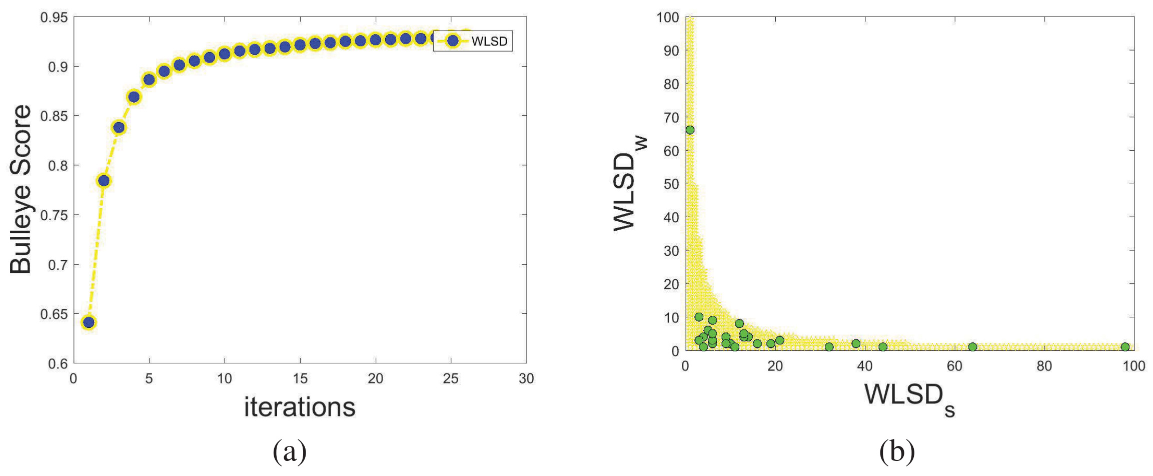


Fig. 2.11 (a) The bulleye score variation with iteration increasing on Tari dataset; (b) The distribution of the selected scales of WLSD in Tari.

Table 2.3 Bull'eye score on rotated and scaled Tari dataset

| Transform | Rotation | | | | Scale | | | |
|---|---|---|---|---|---|---|---|---|
| | 0° | 30° | 60° | 90° | 1 | 0.5 | 1.5 | 2 |
| **WLSD** | 93.09 | 93.27 | 93.64 | 93.12 | 93.09 | 93.65 | 93.02 | 92.83 |
| **CAF** | 87.35 | 88.86 | 88.76 | 87.77 | 87.35 | 89.09 | 87.67 | 86.35 |

We also find that some scores on transformed dataset are even higher than the original, which may because of the de-noising effect caused by image transform.

### 2.5.3 Experiment for WLSD based Tracking

In this section, WLSD is applied in infrared object tracking and tested in infrared tracking videos under background clutters. Considering the infrared targets lack the texture information, as well as the tracker efficiency, we compare the proposed multi-feature integration tracking method with mean-shift [88] tracker. Mean-shift is a widely used tracking algorithm, and is effective in tracking targets of both the high or low resolutions.

#### Implementation Details

The implementation details of the proposed method in Sec. 2.4.2 are discussed in this section. All the region growing method used in target segmentation is based on 8-connected neighborhood. Since most of the infrared target and its background present two peaks under the

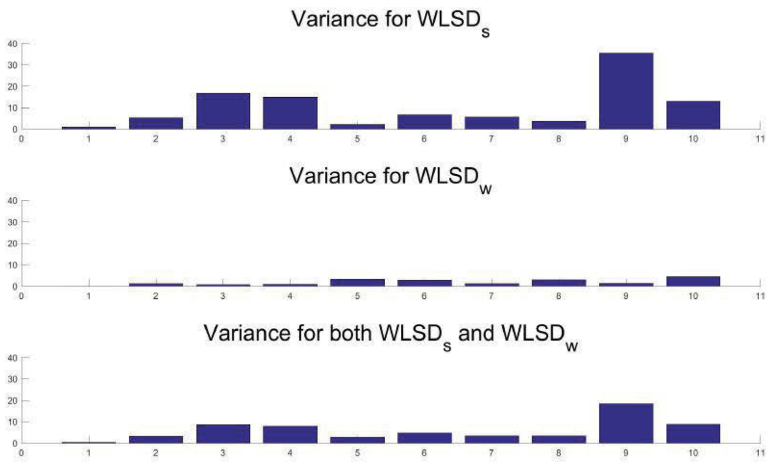


Fig. 2.12 The variance of the first ten selected scales on Tari dataset.

histogram for the gray scales, we evaluate the histogram for the extended patch with 1.5 times of the target size. Assuming $H_h$ and $H_l$ are the maximum and minimum of the histogram, we set the adaptive segmentation threshold to $T = (H_h + H_l)/2$. In addition, the kernel size for local maxima detection is set to $n = 5$, and the location constraint parameter is set to $T_l = 2v$, where $v$ is the target velocity. The area constraint $T_a$ is set to $0.2A_{t-1}$, indicating the area of the target candidate should not be larger than 0.2 times of the target size detected in the last frame. We extract the WLSD feature of the target candidates that satisfy the area constraint. According to the scale distribution experiments on MPEG-7 and Tari datasets, we use $WLSD_{10\rightarrow 3}$ as the whole shape descriptor and $WLSD_{3\rightarrow 3}$ to focus on the local shape variations. We extract 50 points for small target and 100 points for big target. Here, we focus on testing the proposed WLSD feature for tracking, and set the target searching parameters manually. However, more advanced target segmentation, searching strategy, or adaptive parameter design can be further exploited.

### Tracking Experiments

1. **Target without occlusions.** In this section, we test the proposed tracking method in a infrared video without target occlusion, where there exists target deformations. The test results are shown in Fig. 2.13. From the Fig. 2.13, it can be seen that after the target suffers from a series of deformations, mean-shift tracker is prone to drift, while the proposed method can track the target accurately since it is based on target segmentation, at the same time, WLSD is capable of discriminating the deformable target from the false alarms.

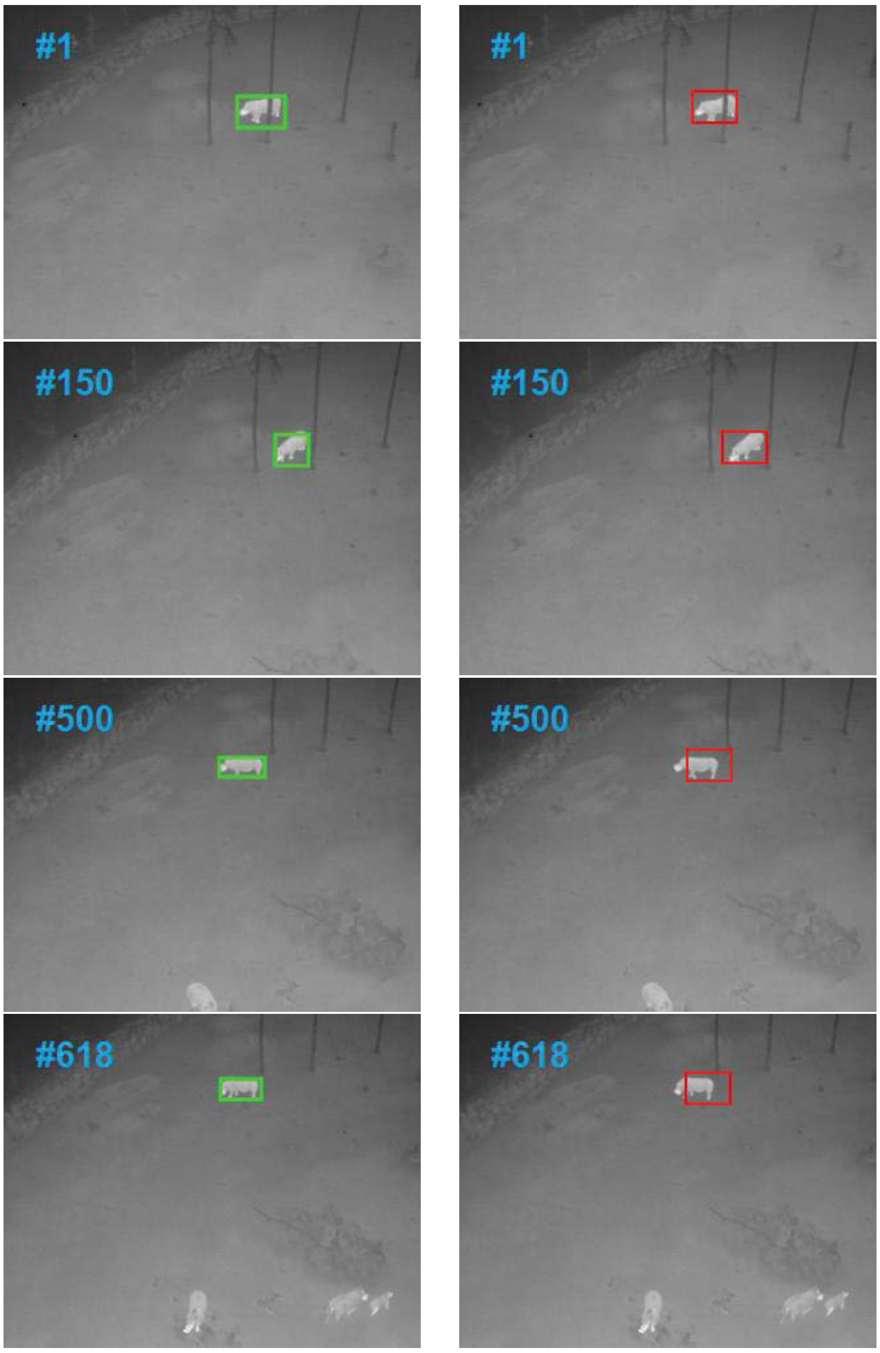


Fig. 2.13 Tracking snapshots for infrared video without occlusion. (Left columns with green rectangles: the proposed method, right column with red rectangles: mean-shift.)

2. **Target with occlusions.** We further test the proposed method using the infrared video that has target occlusions. Fig. 2.14 illustrates that a plane flies behind the power tower, with our method denoted by green box and the compared method by red box. Because of the low resolution of the target, the tracker tends to drift easily. As shown in Fig. 2.14, when the plane flying through the first power tower, mean-shift algorithm fails to track the target gradually and totally loses the target after frame 270. As the gray scale of the target and background is similar, it's hard to discriminate them. While the proposed method adopt multi-feature integration including target shape, trajectory, which is capable of tracking the target with low resolution and fast motion.

## 2.6 Conclusion

In this chapter, we propose a new shape descriptor based on Weber's Law, named Weber's Law Shape Descriptor (WLSD) for high efficiency infrared object tracking. The key idea of WLSD is to capture the salient variations of a shape contour that stimulate human perception. We first design Contour Angular Feature (CAF) as the original stimulus intensity and then use CAF to construct the WLSD feature according to Weber's Law. Furthermore, WLSD is extended to multi-scale to improve its description capacity. Finally, feature selection algorithm is applied to the multi-scale WLSD to extract its discriminative scales. In addition, we apply WLSD to infrared object tracking and propose a multi-feature integration tracking framework, which fuses object shape, area and trajectory features. The proposed WLSD is first tested on two public available shape datasets (MPEG-7 and Tari) to demonstrate its efficiency in shape feature extraction, and then the proposed tracking framework is validated on infrared tracking videos to show its effectiveness.

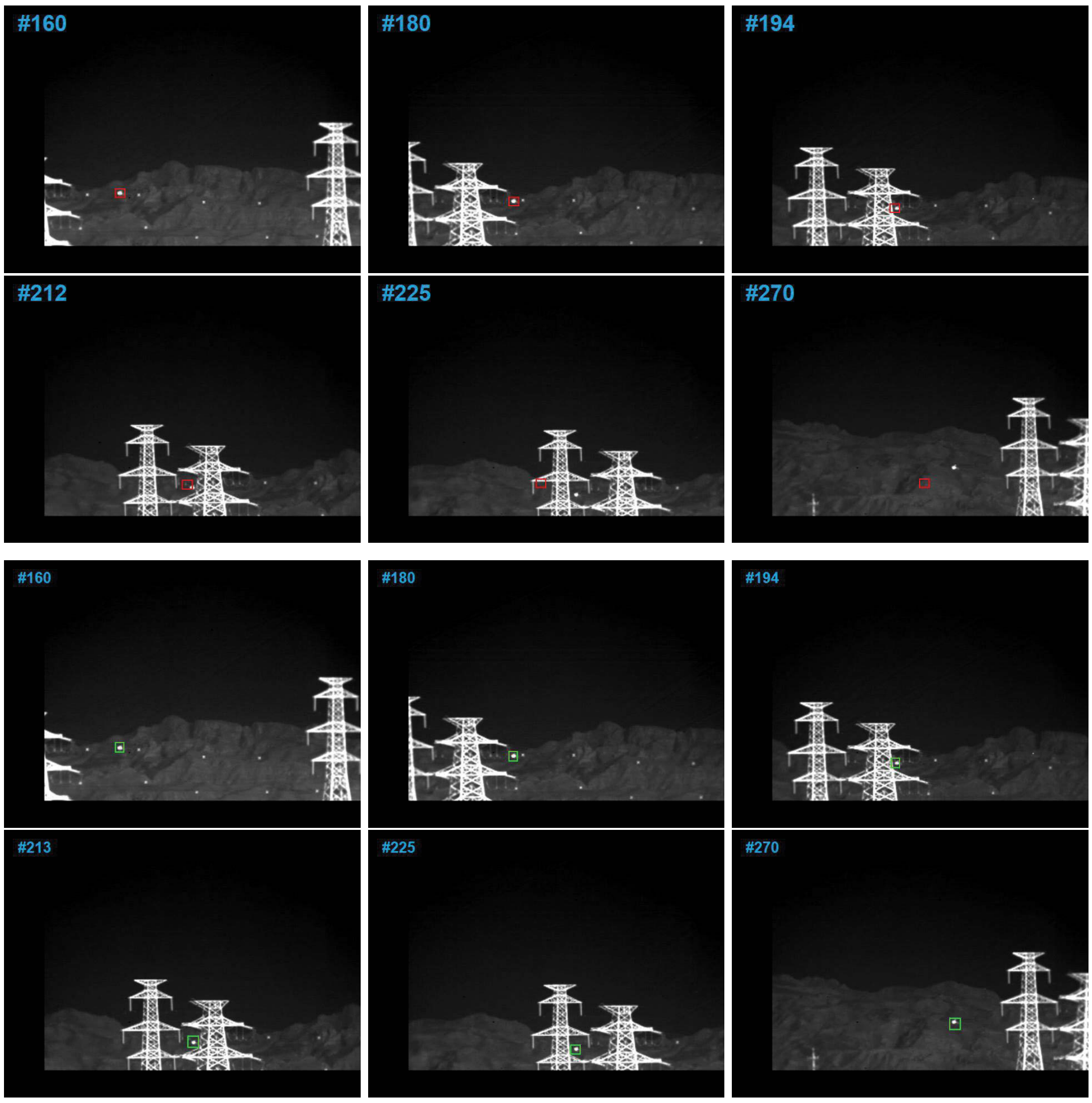


Fig. 2.14 Tracking snapshots for infrared video with occlusion. (Lower two rows with green rectangles: the proposed method, upper two rows with red rectangles: mean-shift.)

# Chapter 3

# Time Varying Metric Learning for Object Tracking

## 3.1 Introduction

In the last Chapter, the proposed shape feature is robust to shape contour deformations, however, it omits the texture information within the shape silhouette. In addition, the proposed method needs the pre-processing algorithm to extract the shape contour, therefore, it is more suitable for the infrared objects that usually have salient shape contour. In the RGB color videos, it often requires to explore more information contains in the object texture. In recent years, there has been a lot of progress in the RGB or gray level imaging representation and the corresponding tracking methods. These include: Haar feature [89] based trackers [8, 10, 27], keypoint (like SIFT, Scale Invariant Feature Transform) feature [77] based trackers [90, 91], HOG (Histogram of Oriented Gradient) feature [78, 92] based tracker [93], and color name [94] feature based tracker [33] etc. In addition, tracking framework is one of the most important techniques. In recently years, tracking-by-detection (TBD) is a very popular tracking framework [2], which adopts the detection-like method to search the target around the target location of the last frame. The TBD methods usually maintains a target online model and the update the model in order to adapt to the object appearance changes [3].

Most of the above existing tracking methods employ a pre-specified metric during the entire tracking process. For instance, Euclidean metric is most commonly used for feature comparison and chi-square distance for calculating histogram distances. If we can obtain features that are discriminative enough from the distracters, the result is satisfactory in many instances. However, under many circumstances, we often observe that a candidate with the closest match by using pre-specified metric do not always turns out to be the true target-of-interest. Thus finding an appropriate metric is necessary.

Metric learning based tracking methods aim to learn the appropriate metric to better handle the classification between object and the background [95, 96, 97]. Tsagkatakis and Savakis [97] combine the online metric learning method with nearest neighbor classification to boost the tracking performance. Jiang et al. [96] propose an adaptive metric learning tracking method, where its goal is to find the best extended nearest classifier to maximize the expected number of training data that are correctly classified. Traditional metric learning based methods treat the object and background as binary classification. However, in the real tracking scenarios, there is no need to exert constraint on the entire background patches as one negative class. Imagine that there are two or more background distracters which are distant apart from each other in feature space, then we must learn a classifier that is capable of distinguishing the target from mixture of clusters in the negative class. To avoid this unnecessary complication, in this chapter, we use the side information that presents a set of pairwise constraints on training data: equivalence constraints that include pairs of “similar" data and inequivalence constraints that include pairs of “dissimilar" data, which can omit background clustering.

Our key motivation is that, in a tracking scenario, it is unnecessary to assume the classes of background. Therefore, it is desirable to provide the so-called “side information" to indicate which data are “similar” or “dissimilar”. This side information based metric learning was first introduced in [98]. Furthermore, Yang et al. [99] extend the work and prove that it performs effectively in the Bayesian learning framework with limited training data. In this chapter, we introduce the side information based metric learning method into object tracking, and show that it can effectively promote the tracking performance.

## 3.2 Related Work

Many trackers train an online binary classifier to distinguish the object from background. One representative method is Support Vector Tracker [6], which uses the idea of SVM combining with optical flow to enhance the performance. Hare et al. [10] further extend the work to Structured SVM to learn the samples with structured labels, which achieves promising results. Babenko et al. [8] introduce multiple instance learning to collect positive and negative samples into bags to learn a discriminative classifier, so as to overcome the drift problem. Zhang et al. [27] adopt the random projection method to reduce the feature dimension which achieves real-time tracking.

Apart from the binary classifier method, Incremental subspace learning and boosting methods are also introduced to the online tracker. Ross et al. [12] propose an incremental learning method for object tracking based on PCA representation. This method can efficiently learn and update a low-dimensional subspace which is composed by PCA eigenvectors. Boosting-based appearance models have been widely used for object tracking [28, 100, 29, 101] due to their efficient discriminative learning capabilities. Practically, discriminative haar-like features are selected, and weak classifiers are correspondingly generated and pooled together as a strong classifier for object location. Grabner et al. [28] demonstrate that an online boosting-based feature selection and classification method can improve the tracking performance dramatically than off-line classifier based algorithms. In [102], an efficient instance probability optimization based feature selection scheme was exploited for better tracking performance with low computational complexity.

Multiple models or trackers are further adopted to obtain more robust performance. Wei et al. [15] propose to combine generative model with discriminative model to jointly handle complex tracking conditions. Kwon and Lee [13] enrich the Bayesian tracking model to multiple state so as to adapt to complicated tracking scenarios. Kalal et al. [7] address the long term tracking problem by using dual experts to handle the positive and negative distracters, which achieves excellent result. In [23, 55], multiple trackers are integrated to form a tracker ensemble to enhance the tracking performance.

Our work draws on the advantages of tracking-by-detection methods but with unique differences. Firstly, our method trains the samples with side information constraint, which focuses on discriminating the object from background by omitting the negative sample clustering. Secondly, unlike the gradient based metric learning methods [97, 96], our method estimates the metric accurately with limited samples in the Bayesian framework, which is more suitable for the object tracking.

The rest of the chapter is organized as follows: in Sec. 3.3, we first introduce the Wishart Process and side information constraint, and then propose our Time Varying Metric Learning model with its SMC solution. In Sec. 3.4, we explain how to apply the proposed model to object tracking. In Sec. 3.5, model validation is conducted on synthetic data, then the proposed tracker is evaluated in the 50 sequences tracking benchmark. Finally, we conclude the chapter.

## 3.3 Time Varying Metric Learning Model

In this chapter, we aim to learn a distance metric to calculate the distance between two feature vectors $\mathbf{x}$ and $\mathbf{y}$, which can be defined as:

$$d_M(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|_M = \sqrt{(\mathbf{x}-\mathbf{y})^T M(\mathbf{x}-\mathbf{y})}, \tag{3.1}$$

where $M$ is a positive semi-definite (PSD) matrix.

### 3.3.1 Wishart Process

Wishart Process [103] is a stochastic process which is able to generate a sequence of random PSD matrices $M_1,...,M_t$ over the time. According to the definition of Wishart Process, the relationship between $M_t$ and $M_{t-1}$ is defined as:

$$\begin{aligned} M_t|v,S_{t-1} \sim & Wishart(v,S_{t-1}) \\ where \quad S_{t-1} = & v(\frac{1}{v}A^{\frac{1}{2}})(M_{t-1})^d(\frac{1}{v}A^{\frac{1}{2}})^T, \end{aligned} \tag{3.2}$$

where $M_t$ is the PSD matrix at time $t$, and $Wishart(v,S_{t-1})$ is the Wishart distribution parameterized by $v$ and $S_{t-1}$, which are the number of degrees of freedom and the time-dependent scale parameter respectively. $A$ is a positive definite symmetric parameter matrix that is decomposed by Cholesky decomposition as $A = (A^{\frac{1}{2}})(A^{\frac{1}{2}})^T$. $d$ is the scalar parameter.

Wishart Process is able to model the dynamic behavior of a set of PSD matrices across time. The scale parameter $S_t$ not only defines the time variation of the PSD matrix but also ensures the proposed matrices are positive definite. The parameters $A$ and $d$ control the variation behavior of the PSD matrices. $A$ is interpreted as revealing how each element of PSD matrix depends on the previous PSD matrix. While parameter $d$ denotes the overall strength of the metric evolution relationship. $d$ is proved to be theoretically bound between $(-1,1)$ [103]. In practice, $d$ is usually within $(0,1)$. More details about Wishart Process and its parameter interpretation can be referred to [103, 104].

### 3.3.2 Side Information Constraint

As mentioned above, we aim to learn a metric to better distinguish positive (target) samples from negative (background) samples by putting no constraint to the negative training data. To achieve this goal, we propose to use pair-wise data constraint instead of treating all the training data as two classes. As a result, by setting the similar pair-wise constraint to positive data, and the dissimilar constraint between pair-wise positive and negative data, we can omit the negative clusters and put all the energy in discriminating the positive data from the negative. Therefore, given a metric, it is necessary to define a probability to describe the similar and dissimilar relationship for data pairs.

Motivated by [99], the probability for two data points $\mathbf{x}_i$ and $\mathbf{x}_j$ to form a similar or dissimilar set under a given distance metric M can be defined as:

$$\begin{aligned} P(c_{i,j}|\mathbf{x}_i,\mathbf{x}_j,M,\mu) &= \frac{1}{1+\exp(c_{i,j}(\| \mathbf{x}_i-\mathbf{x}_j \|_M^2 -\mu))} \\ where \quad c_{i,j} &= \begin{cases} +1 & (\mathbf{x}_i,\mathbf{x}_j)\in S \\ -1 & (\mathbf{x}_i,\mathbf{x}_j)\in D, \end{cases} \end{aligned} \tag{3.3}$$

where $S$ and $D$ denote the similar and dissimilar set respectively. $\mu$ is the threshold, which means that two data are more likely to be in the same constraint set when their distance is less than the threshold $\mu$. Note the above formula is a discrete distribution, whose sum over random variable $c_{i,j} \in \{+1,-1\}$ equals 1. Assuming each pair of the training data is independent, the whole likelihood probability for all the constraints in S and D is define as:

$$\begin{aligned} P(S,D|M,\mu) = &\prod_{(i,j)\in S} \frac{1}{1+\exp(\| \mathbf{x}_i-\mathbf{x}_j \|_M^2 -\mu)} \\ \times &\prod_{(i,j)\in D} \frac{1}{1+\exp(-(\| \mathbf{x}_i-\mathbf{x}_j \|_M^2 -\mu))} \end{aligned} \tag{3.4}$$

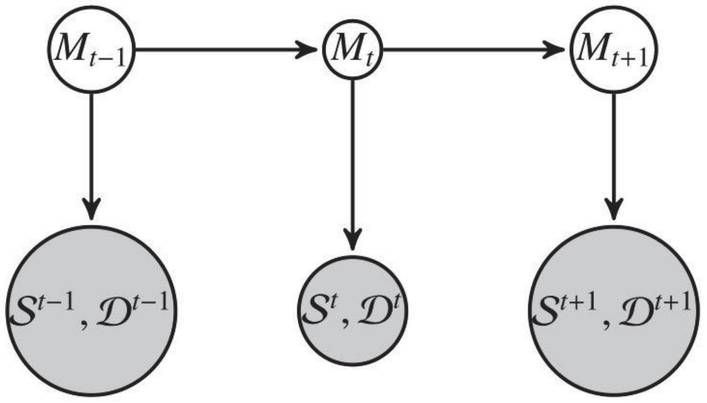
### 3.3.3 Graphical Model for Time Varying Metric Learning

Unlike most existing metric learning frameworks which learn the data from a single time instance, we introduce a Bayesian framework to learn the metric recursively. Given a set of labelled "similar" and "dissimilar" constraints for training data at each time step $t$, we propose the Time Varying Metric Learning (TVML) graphical model as illustrated in Fig. 3.1. The TVML likelihood probability $P(S_t,D_t|M_t)$ is defined as Eq. 3.4, whereas the transition probability $P(M_t|M_{t-1})$ is naturally defined as the Wishart Process shown in Eq. 3.2.

The proposed TVML model is capable of estimating the metric behaviour that is well suited for distinguishing the similar data from the dissimilar ones while minimizing the sum of distances in the similar set. At each time step, the prior and the likelihood of the model interacts. As the prior carries forward its state information from the previous time step, TVML needs only a few data to estimate the current state $M_t$.

### 3.3.4 The Sequential Monte Carlo Solution

After the TVML model is a established, our aim is to estimate metric $M_t$ at each time step given all the previous equivalence and inequivalence knowledge, i.e. $P(M_t|S_{1:t},D_{1:t})$.



$M_{t-1}$
$M_t$
$M_{t+1}$
$\mathcal{S}^{t-1},\mathcal{D}^{t-1}$
$\mathcal{S}^{t},\mathcal{D}^{t}$
$\mathcal{S}^{t+1},\mathcal{D}^{t+1}$

Fig. 3.1 Time Varying Metric Learning Graphical Model

The above issue can be viewed as a filtering problem, we use the Sequential Monte Carlo method (SMC) [17, 105] to solve it. According to SMC method, the posterior distribution is represented by the following Monte Carlo approximation:

$$\hat{P}(M_t|S_{1:t},D_{1:t}) \propto \sum_{i=1}^{N} P(S_t,D_t|\hat{M}_t^i)\delta_{\hat{M}_t^i}(m_t) = \sum_{i=1}^{N} w_t^i \delta_{\hat{M}_t^i}(m_t) \tag{3.5}$$

where "ˆ" is the estimation symbol and the particles weighted by $P(S_t,D_t|\hat{M}_t^i)$ are used to approximate the posterior at each time $t$. The proof is detailed at the end of this section. In every recursive step, we use the transition $P(M_t|M_{t-1})$ as the proposal to propagate the particles. Finally we conduct the resampling step to overcome the degeneracy problem.

As shown in [17], when the variance of the particles is small, the resampling step might be unnecessary. Therefore, in this chapter, the SMC solution conducts the resampling step only when the variance of weights is larger than the pre-specified threshold. This is assessed by the so-called Effective Sample Size (ESS): $ESS = \left(\sum_{i=1}^{N}(w_t^i)^2\right)^{-1}$. The ESS varies between 1 and N and resampling is triggered only when it is below a threshold $N_T$, typically $N_T = N/2$ [17].

**Proof:** The posterior $P(M_t|S_{1:t},D_{1:t})$ satisfies the Baysian recursion process

$$P(M_t|S_{1:t},D_{1:t}) \propto P(S_t,D_t|M_t)P(M_t|S_{1:t-1},D_{1:t-1}) \tag{3.6}$$

where

$$\begin{aligned} &P(M_t|S_{1:t-1},D_{1:t-1}) = \\ &\int P(M_t|M_{t-1})P(M_{t-1}|S_{1:t-1},D_{1:t-1})dM_{t-1} \end{aligned} \tag{3.7}$$

The above equations provides a recursive solution to the model. According to Monte Carlo assumption, the posterior can be represented by a set of unweighted particles:

$$\overline{P}(M_{t-1}|S_{1:t-1},D_{1:t-1}) = \sum_{i=1}^{N} \frac{1}{N}\delta_{\overline{M}_{t-1}^{i}}(m_{t-1}) \tag{3.8}$$

where " – " denotes the unweighted estimation to distinguish from the weighted estimation "ˆ" in Eq. 3.5. Consequently, the recursive posterior can be represented as:

$$\overline{P}(M_t|S_{1:t},D_{1:t}) \propto P(S_t,D_t|M_t)\sum_{i=1}^{N} P(M_t|\overline{M}_{t-1}^{i}) \tag{3.9}$$

Then each particle is proposed by the proposal distribution $\hat{M}_t \sim P(M_t|\overline{M}_{t-1}^{i})$. Finally, by taking it into into Eq. 3.9, Eq. 3.5 is obtained.

## 3.4 TVML for Tracking

In this section, we show how to apply the proposed TVML model into object tracking. We first elaborate the proposed tracking framework, and then introduce the target representation method and the online model update scheme. Finally, the algorithm flow is summarized individually.

### 3.4.1 TVML Tracking Framework

As shown in Fig. 3.2, the whole tracking process is divided into three main steps: training, testing and model update.

**Training**: Given the previous target location, a set of image patches are collected as training data. Image patches close to the target area are considered as positive samples, while the image patches far away are treated as negative samples. This collection method is quite useful especially for overcoming the target distracting problem. As the learned metric can well distinguish the true target from the distracters before they interact or occlude. After collecting the positive and negative image patches, feature extraction is conducted to all the patches, which is discussed in the Sec. 3.4.2. Then the extracted feature vectors are fed into the TVML model to learn the metric for testing.

The key point of training step lies in the training constraint. Unlike traditional methods that only exert two class label to the training data, our method utilize the side information that constrains each pair of the positive samples as similar, i.e., having smaller distance, and each pair of positive and negative samples as dissimilar, i.e., having larger distance. The method

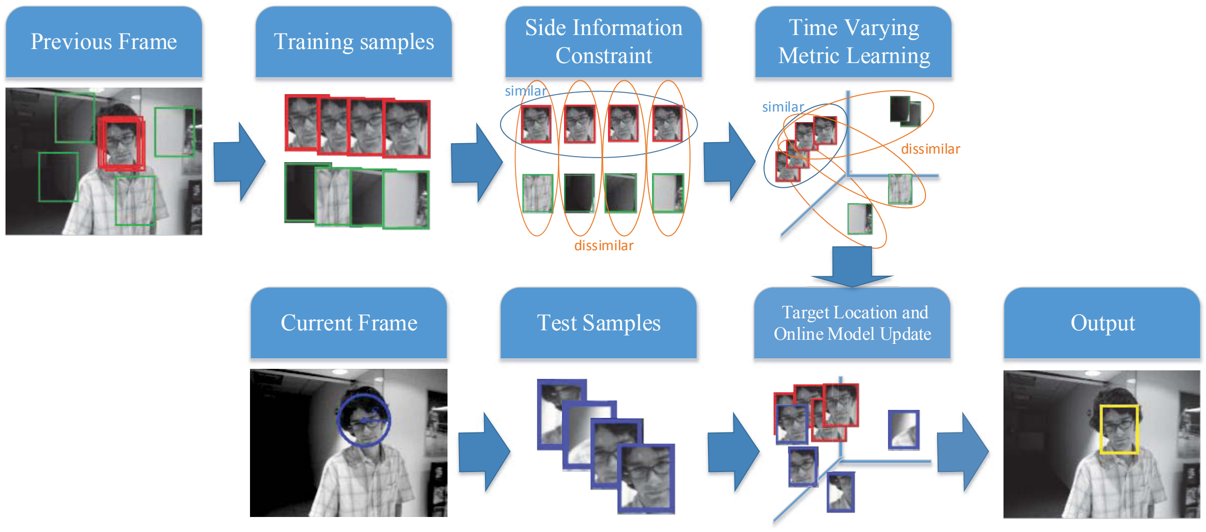


Fig. 3.2 TVML for tracking. There are three main steps: training (the flows in the first row), testing and model update (the flows in the second row). The training step collect the training samples and learn the metric. The testing step then use the learned metric to search the current target location followed by the step of online model update.

does not place any constraints over pairs of negative samples, as it concentrates its entire effort in distinguishing the target from background.

**Testing**: Instead of searching for the target with smallest distance to the previous object region, we adopt a collection of previous target appearances to form a template library for online model [97], so as to adapt to the object appearance variation. Using the template library and the learned metric, the location of the target for the current frame is obtained by locating the region that has the smallest distance with the template library.

Finally, we update the model as described in Sec. 3.4.2.

## 3.4.2 Target Representation and Model Update

Target Representation, also known as feature extraction, is a critical step in object tracking. In most trackers, low level features, such as Haar-like features or gray-scale features are adopted [8, 27, 4]. To handle the target appearance more effectively, we adopt the dense Scale Invariant Feature Transform (SIFT) [106] to represent the target. SIFT feature is famous for its scale and rotation invariance, and it is also invariant to small changes of the object appearance. It has been further shown that extracting SIFT feature on the regular grid, also called dense SIFT, is capable of outperforming that obtained by keypoint detection [107]. In this chapter, we adopt the dense SIFT feature to represent image patches.

The bottleneck of the dense SIFT feature is its computation load because of its high dimension, since each grid point on the image patch will generate a 128-dimension feature vector. To overcome this problem, we adopt random projection as in [97] to reduce the feature dimension. Random Projection (RP) is a data-independent method for dimension reduction, which do not needs the prior knowledge of the object appearance.

To tackle the target appearance variation issue, we introduce an online model to adapt to appearance changes similar to [97]. The online model maintains a template library consisting of previous target appearances, and update the library by replacing old templates with new ones if necessary. Let the feature vector of the image patch denoted by $\mathbf{x}$, and the template library containing feature vectors of the past target appearances denoted by $\mathbf{L} = \{\mathbf{x}_1, \mathbf{x}_2, ...\}$. The distance between a target candidate $\mathbf{x}$ and the template library is defined as the minimum distance of $\mathbf{x}$ to each of the template element in $\mathbf{L}$ given the metric $M$, i.e. $d = \min_{\mathbf{x}_i \in \mathbf{L}} d_M(\mathbf{x}, \mathbf{x}_i)$.

More details of the above strategies can be referred to [97, 108].

Based on the above discussion, the algorithm flow is summarized as follows.

**Algorithm 2:** TVML Tracker

```
input  : Initial target state $\mathbf{s}_1$
output : Estimated target state $\mathbf{s}_t = (\hat{x}_t, \hat{y}_t)$
repeat
    Given target state $\mathbf{s}_{t-1} = (\hat{x}_{t-1}, \hat{y}_{t-1})$, collect training data from frame $t-1$;
    Set the side information constraint to $S_{t-1}$ and $D_{t-1}$ according to Sec. 3.4.1;
    Learn metric $M_{t-1}$ using SMC as indicated in Eq. 3.5;
    Update the online model $\mathbf{L}_{t-1}$ according to Sec. 3.4.2;
    Estimate the target state $\mathbf{s}_t = (\hat{x}_t, \hat{y}_t)$ in frame $t$ using metric $M_{t-1}$ and online model
    $\mathbf{L}_{t-1}$;
until Last frame of video sequence;
```

## 3.5 Experiments

The whole experiment is divided into two parts. Firstly, we validate the proposed model using synthetic data. Secondly, we evaluate our proposed model in the 50 sequences tracking benchmark [2], and compare its performance with the state-of-the-art methods. The experimental platform is on a 3.20GHz CPU with 8GB RAM.

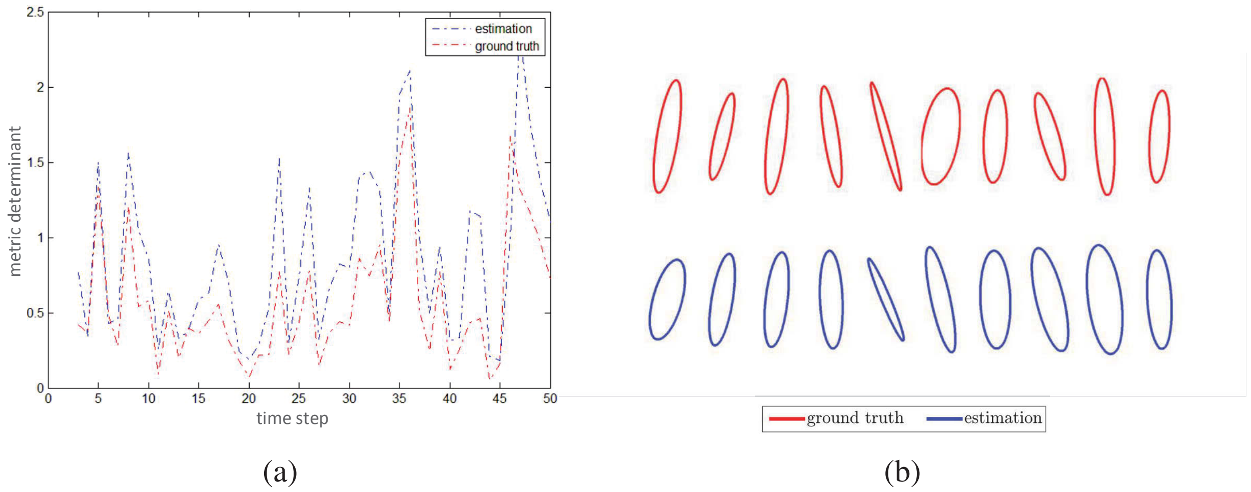


Fig. 3.3 TVML on synthetic data: (a) metric determinant; (b) elliptical representation.

### 3.5.1 Model Validation

In order to demonstrate the effectiveness of our TVML model, we first use the synthetic data to validate that it is able to catch various metrics. We generate synthetic data by sampling the likelihood probability shown in Eq. 3.4. For efficiency and simplicity, we choose the Metropolis-Hasting sampling method [109]. To validate the generalization of our model, we set the parameters of Wishart Process for ground truth and test model differently, where the parameters of the former are $v = 5, d = 0.3, A = I$, and those of the latter are $v = 9, d = 0.5, A = I * 0.8$ ($I$ is the identity matrix). In order to illustrate the experimental results clearly, we set the data dimension to 2, so as to plot the covariance matrix as the ellipse [104, 25].

Fig. 3.3 (a) shows the determinant of the metric for both the ground truth and its estimation, and Fig. 3.3 (b) shows the randomly selected time-dependent elliptical representations, which is more accurate to illustrate the estimation and ground truth. It can be seen that the trend of estimated metric determinant is very close to the ground truth even their parameters are very different. The figure for elliptical representation further illustrates the estimation accuracy of the mode. The above results demonstrate that TVML can capture the metric both from the trend and precision. In addition, the estimated matrix determinant in Fig. 3.3 (a) is overall larger than the ground truth because of the monotonicity of the likelihood function. It is worth noting that the likelihood monotonicity is more reasonable since it is better for the distances among the $S$ set smaller and $D$ set larger.

## 3.5.2 Experiment for Object Tracking on OTB-50

In this section, the proposed tracker is evaluated on the 50 sequences object tracking benchmark (OTB-50) [2]. The compared methods are the 27 trackers provided by the benchmark. In the following sections, we firstly describe the implementation details of the proposed TVML tracker and the evaluation criteria, and then report the experimental results for further evaluation.

### Experiment Setup

In this chapter, we extract dense SIFT feature in the resized $45*45$ image patch with $2*2$ features spread in the patch uniformly. The feature dimension of one image patch is then $128*4 = 512$. We then use RP to reduce its dimension to 128.

As indicated in Sec. 3.3.1, the parameter $A$ and $d$ play an important role in determining the dynamic behavior of the covariance structure. The parameter $d$ indicates the overall strength of matrix evolution. $d$ close to 0 means a weak overall effect of the current volatility on future values, and $d$ close to 1 indicates high persistence. In tracking scenario, the target and background changes are relatively continuous and stable. Therefore, we make a trade-off to set $d = 0.5$. Parameter $A$ reveals how each element of the PSD matrix depends on the elements of the previous PSD matrix. As it's hard to predict the weight relation between the elements of the feature vectors, we do not give too much pre-defined constraint and set $A = I*1.2$. Therefore, the parameters of the Wishart Process are set to $d = 0.5, A = I*1.2, v = 4.5$, where we find the setting of $v$ has little influence of the performance in practice. The parameter $\mu$ of likelihood is evaluated from the first frame, which is set to the mean of the median value of $S$ set and $D$ set. The number of the positive and negative training examples are 15 and 20 respectively. For the SMC solution, the particles are initialized as the identity matrix and are propagated according to the Wishart Process. Resampling is conducted when the ESS is below the threshold of $N/2$, where $N$ is the number of particles. In this chapter, $N$ is set to 200. To keep a rich representation for the object, the template library contains five templates elements. To tackle the target occlusion issue, we set a pre-defined threshold $\xi$. If $d_t > \xi$, then the target occlusion is detected, and the model update pauses temporarily to prevent bad update. The algorithm is implemented in Matlab & C. The code has not been optimized and runs at roughly $1 \sim 3s$ per frame at the current state.

Two performance evaluation metrics are used in the experiment: Overlap Success Rate (OSR) and Center Location Error (CLE). The OSR is defined as $score = \frac{ROI_T \bigcap ROI_G}{ROI_T \bigcup ROI_G}$, where $ROI_T$ is the area of the estimated bounding box and $ROI_G$ is area of the ground truth bounding box. CLE is the center distance between the tracking result and the ground truth bounding box.

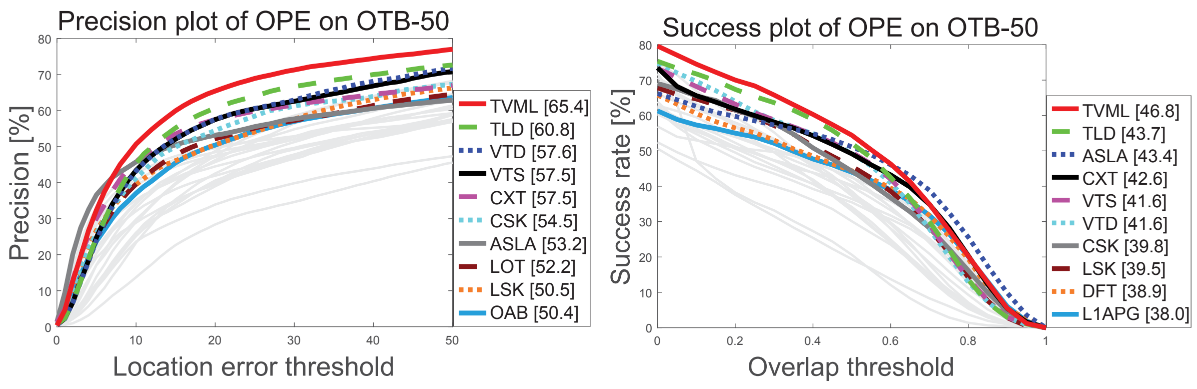


Fig. 3.4 The success plot and precision plot over OTB-50 using one pass evaluation (OPE). The legend illustrates the area under curve (AUC) for the success plot, and the score of the threshold 20 for the precision plot. Only top-10 trackers are colored, and the others are shown in gray curves. Legend of the same color denotes the same rank.

### Overall Performance

The overall performance of the benchmark is illustrated in two plots: precision plot and success plot. The former plot shows the percentage of frames whose CLE is within the given threshold distance. The latter plot shows the ratios of successful frames at the thresholds from 0 to 1. The successful frames are counted if the OSR is larger than 0.5.

The overall performance is shown in Fig. 3.4. For clear illustration, only the top-10 trackers are shown in the plots, and the other trackers are shown in gray curves. From the plots, it is shown that TVML ranks first in both the evaluation methods, and TLD [7] ranks the second. Our method has improved by 7.6% and 7.1% for the precision and success measure respectively compared to the second rank tracker. Note that VTD [13], VTS [23] and ASLA [14] are all based on Bayesian framework, which demonstrate the effectiveness of our tracker. Among the compared trackers, VTD and VTS use the image intensity, color and shape as the combined feature. TLD adopts patch template as features. It demonstrates to some extent that the high level dense SIFT feature performs better than the low level features.

### Method Efficiency Analysis

The computation load of the top-10 trackers for overall performance are shown in Table 3.1. From the table, it is observed that FPS of sampling based method have higher computation load than grid search methods. Particularly, CSK runs much faster than the other methods since it is based on Fast Fourier Transform (FFT). In addition, TLD ranks second in FPS because the whole algorithm is based on gray scale feature, which needs less computations than those of high level features. Our method is mainly based on SMC sampling, and it has relatively the same computation load than the other sampling based trackers (Note the model solver is based

Table 3.1 Computation loads of the top-10 trackers in Fig. 3.4 are presented in three aspects, including frames per second (FPS), tracking method and the implementation. For method, S: Sampling based method, GS: Grid search based method. For implementation, M: Matlab, MC: Matlab + C, E: executable code.

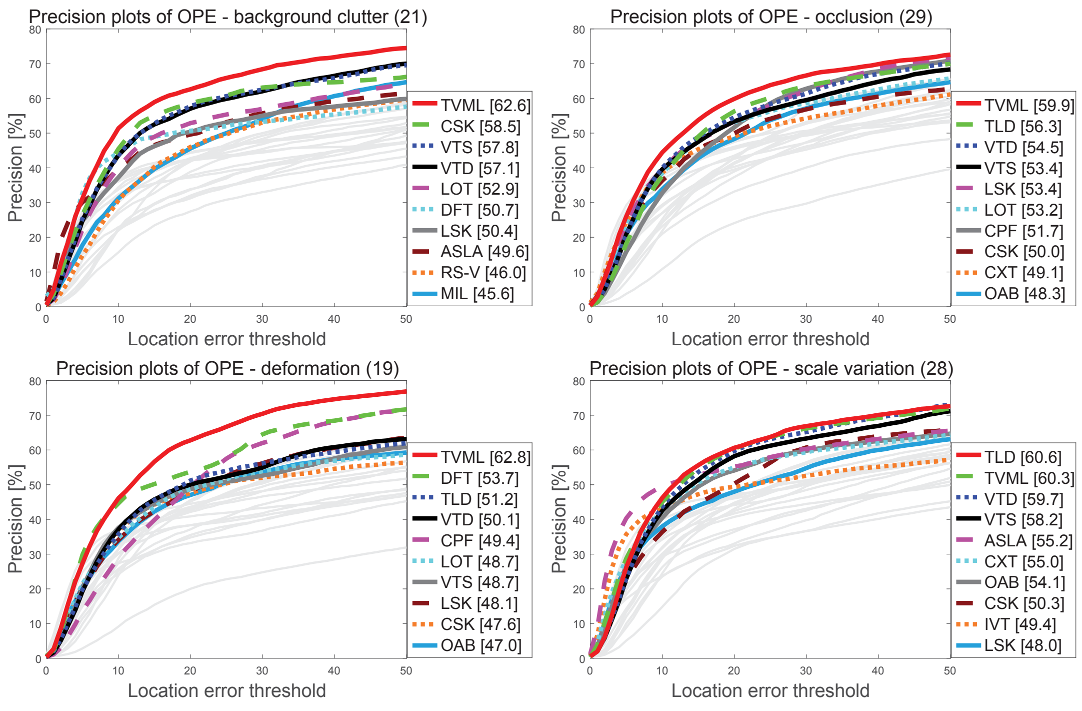
| | **TVML** | **TLD** | **VTD** | **VTS** | **CXT** | **CSK** | **ASLA** | **LOT** | **LSK** | **OAB** |
|---|---|---|---|---|---|---|---|---|---|---|
| **FPS** | 2.56 | 25.74 | 2.75 | 2.72 | 10.67 | 268.45 | 2.48 | 0.47 | 0.072 | 5.13 |
| **Method** | S | GS | S | S | GS | GS | S | S | GS | GS |
| **Implementation** | MC | MC | MC (E) | MC (E) | C | MC | MC | M | M (E) | C |

on sampling, and the search scheme is grid search). Moreover, during the experiments, it is observed that the dense SIFT computation takes about 40% time in our whole algorithm. Taken the whole computation load into consideration, we draw conclusion that the proposed method has normal computation complexity among the sampling based methods.

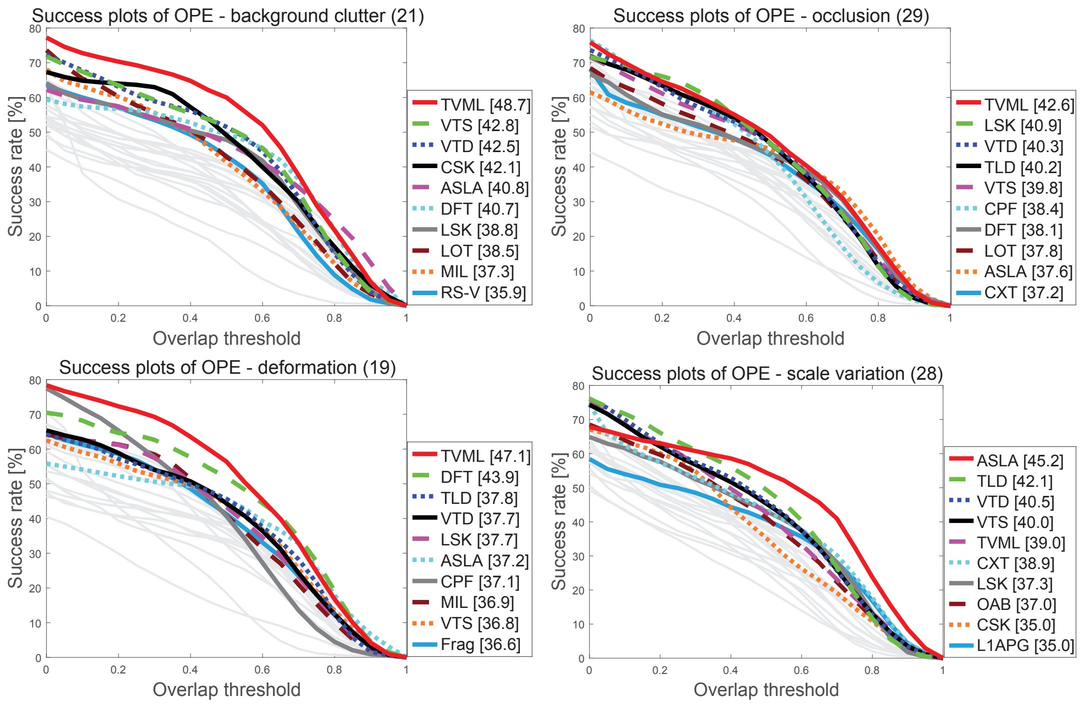
### Attribute-based Performance

All the sequences in the benchmark are then divided into four groups according to their attributes, including background clutter, occlusion, deformation and scale variation. The attributes experimental results are shown in Fig. 3.5. In accordance with our assumption, one of the most effective scenarios handled by TVML tracker should be background clutter. From the plots in Fig. 3.5, it is illustrated that both the precision and the success rate of the proposed method rank first among the compared trackers. To be more specific, in the background clutter attribute sequences, TVML has improved the precision and success score by 7% and 12.8% respectively relative to the second tracker, which demonstrate our method is very effective in handling complex background clutters. In addition, in the other two attributes of occlusion and deformation, TVML also outperforms against other methods. In the scale attribute sequences, TVML ranks second in the precision measure, while ranks fifth in the success rate metric. This is reasonable since all of the trackers of TLD [7], ASLA [14], VTD [13] and VTS [23] have scale estimation. Therefore, their success scores are higher. However, only TLD outperforms TVML in scale attribute measured by precision metric. This is because the robustness of dense SIFT feature extracted on the image grid.

Lastly, the snapshots of some typical sequences of the whole benchmark are shown in Fig. 3.6. Among the eight sequences, *Basketball, Football* and *Singer2* go through severe background clutters. TVML performs well in the above sequences, which demonstrate the proposed metric learning framework is capable of handling these background clutter scenarios. *Bolt, Football* and *SUV* have similar target distracters. In the *Bolt* sequence, most of the trackers drift at the last few frames while TVML can deal with such appearance changes. This also



(a) Precision plots



(b) Success plots

Fig. 3.5 The precision plots and success plots for four main attributes of the benchmark, i.e. background clutter, occlusion, deformation and scale variation. The figure in the title denotes the number of sequences belongs to specific attribute.

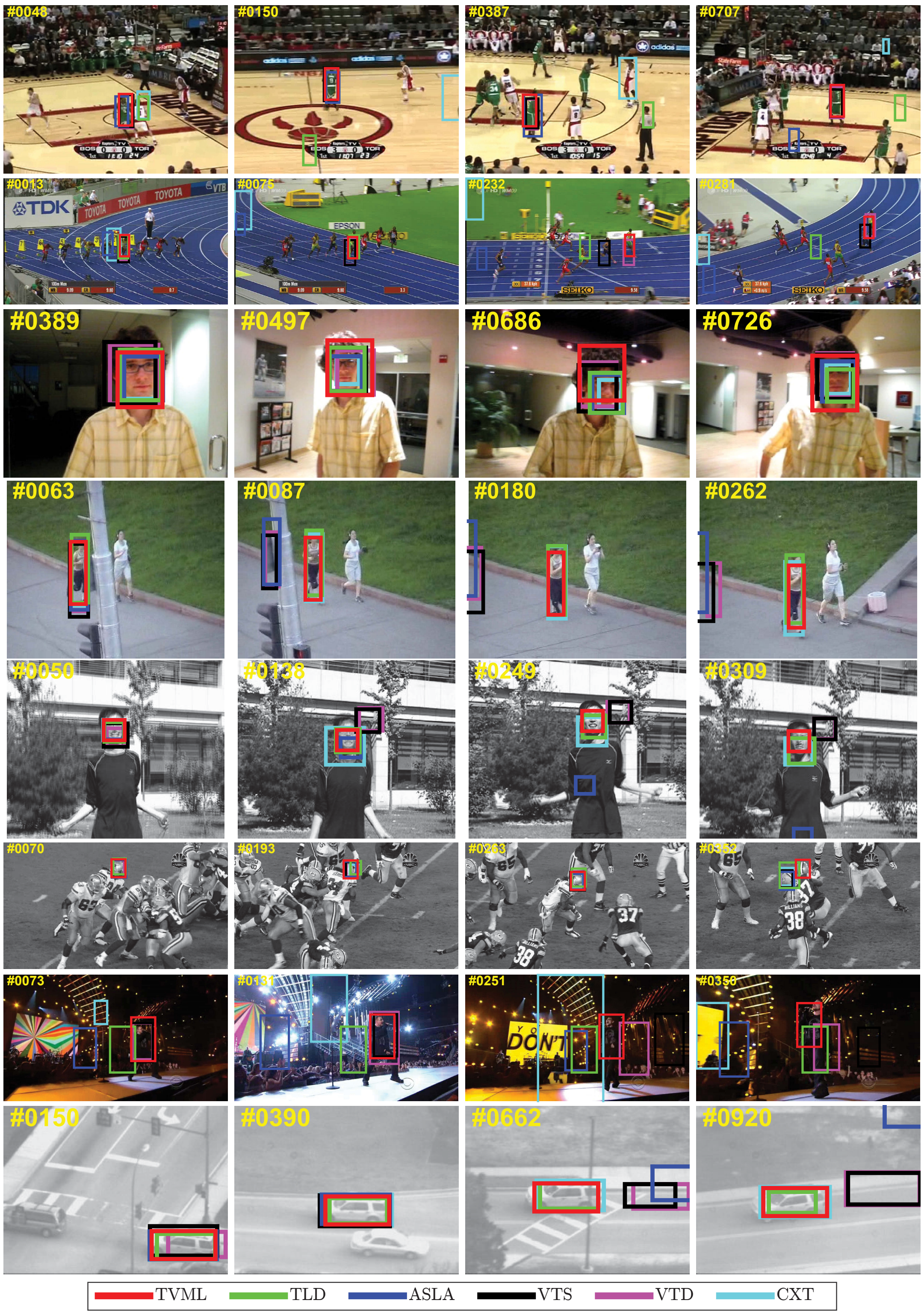


Fig. 3.6 Tracking snapshots of the top six algorithms among the overall performance including TVML, TLD [7], ASLA [14], VTS [23], VTD [13], CXT [110] over eight sequences. The illustration example videos from top-left to bottom-right are *Basketball, Bolt, David, Jogging, Jumping, Football, Singer2 and SUV*.

demonstrate our online model mechanism is effective. At the last snapshot for *Football*, most of the trackers drift to the nearby distracter, but our method is able to catch the true target due to the online metric learning strategy. The same conclusion can be draw from *David*, *SUV* and *Jogging*.In practice, we find that in most cases, the proposed tracker can catch the target even with partial occlusion. This is because of the robustness of the dense SIFT feature as well as the online model.

## 3.6 Conclusion

This chapter proposes a novel Time Varying Metric Learning model for object tracking. The proposed model is under the Recursive Baysian Estimation framework, which is capable of learning the metric with limited number of training data. Wishart Process is introduced as the transition model to capture the dynamic metrics under side information constraint. In addition, we introduce "side information" constraint to train the model that is more suitable for background clutters. The experiments demonstrate the effectiveness of the proposed method.

# Chapter 4

# Tracker Snapshot Based Multi-Expert Tracking

## 4.1 Introduction

Variations of target appearances due to illumination changes, heavy occlusions and abrupt motions are the major factors for tracking failures. A commonly used strategy is to design an online model that evolves forward to adapt to the target appearance changes. The main drawback of this method is that online models tend to drift with the time passing by. The drift happens even more easily due to large appearance changes of the object, abrupt motions and heavy occlusions.

The methods of the last two chapters are all based on single tracker. To tackle the model drift problem, many methods propose to use tracker ensembles which are composed of more than one base trackers to determine the target position [13, 23, 15, 7, 111]. One strategy is to establish a tracker pool and choose the most appropriate tracker each frame to make the best decision [13, 23, 15]. Others use multiple experts working parallel to better discriminate the target from the background [7, 111]. One of the representative work is that in [55], Zhang et al. propose to use the multi-expert restoration scheme to address the model drift problem, where an entropy based loss function is defined to determine whether the current tracker is reliable and should be restored to the historical tracker. The proposed tracking framework adopts online SVM as the base tracker, which shows very robust performance. The work by Zhang indicates that, to some extent, the historical trained trackers, also called tracker snapshots, can be used to prevent the model drift.

The key motivation of our method is the observation that tracking failures can be effectively handled by exploiting the historical tracker snapshots. As shown in Fig. 4.1, during the tracking

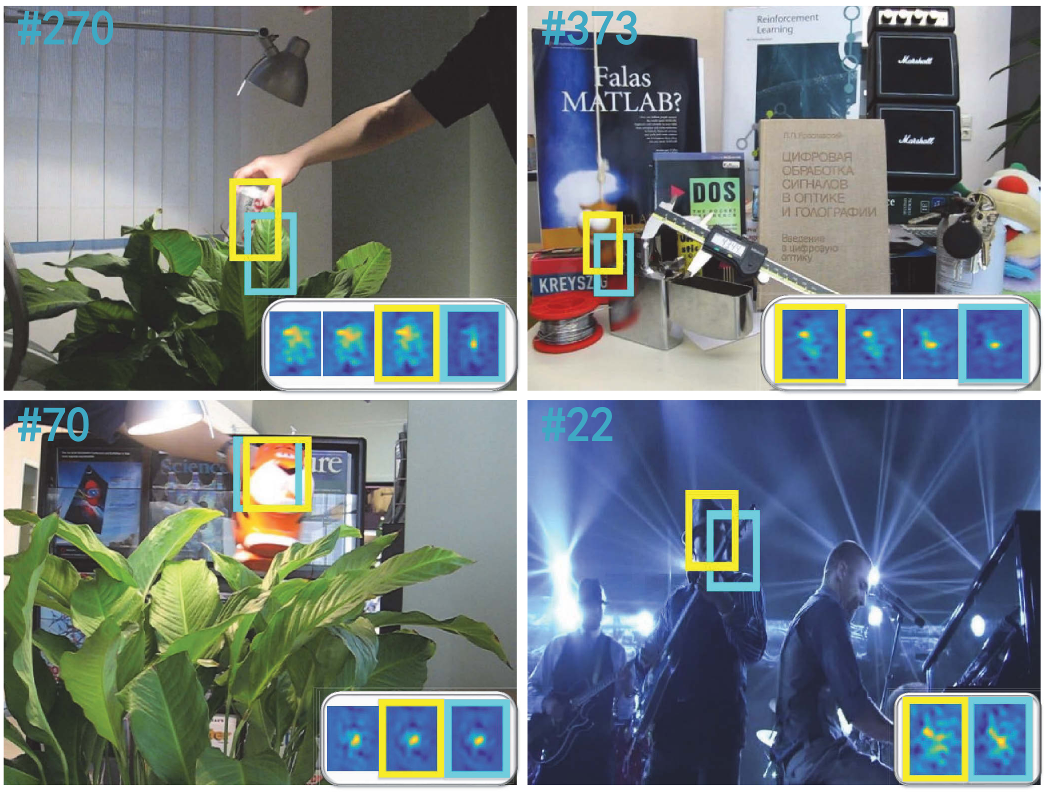


Fig. 4.1 Four typical sequences (*Coke*, *Lemming*, *Tiger1*, *Shaking*) show the importance of exploiting the historical tracker snapshots. The cyan bounding box is the tracking result of our base tracker, while the yellow box is the selected tracker historical snapshot by the multi-expert framework. The tracker snapshots are stored at the pre-defined interval, and the corresponding response maps are illustrated at the bottom-right of the image (listed in chronological order from left to right). The color brightness of the response map indicates the confidence degree of the tracker snapshot. The number at the top-left corner is the frame count.

process, the object goes through significant appearance variation, occlusion and illumination change. Therefore, the current tracker tends to drift (the response map is framed by the cyan box). However, it is observed that the target location can be accurately estimated by most of the historical tracker snapshots. For example, in sequence *Coke*, after the object having been occluded by the leaves, the current tracker is distracted by the background, but its three past snapshots are all able to identify the true target. The same phenomenon is shown in sequence *Lemming* and *Tiger1*. In addition, as illustrated in sequence *Shaking*, we will show that by designing the appropriate snapshot selection criteria, the early period tracking failure can also be avoid.

The above phenomenon gives the main insight of our work. During the tracking process, single tracker is easy to overfit when there is target partial occlusion, abrupt motion, background clutter and other factors lead to object appearance variation. However, the above moments are relatively short during the whole tracking process. On the other hand, the diversity of target appearance is usually limited, and cannot be varying significantly all the time without restore to is past appearance. Therefore, sometimes the past tracker snapshot is capable to recognize the object better than the current tracker. Particularly, the past snapshot can re-identify the target after its occlusion and abrupt motion, which is naturally to rescue the tracking failure.

As a result, in this chapter, we exploit the historical tracker snapshots and show that tracking performance can be effectively promoted by exploiting the relationship between them. We propose a Scale-adaptive Multi-Expert (SME) tracker, which combines the current tracker and its historical trained snapshots to construct a multi-expert ensemble. The best expert in the ensemble is then selected according to their accumulated reliability score. The base tracker estimates the translation accurately with regression based correlation filter, and an effective scale adaptive scheme is introduced to handle scale changes on-the-fly.

## 4.2 Related Work

The work in this chapter belongs to tracker ensemble methods. There are a lot of tracker ensemble based trackers. For example, Kwon and Lee [13] decomposes traditional Bayesian recursive framework into basic models, and uses the MCMC sampling to integrate them. In [23], the proposed method not only samples the target state space but also the tracker space to handle challenging tracking scenes. Kalal et al. [7] address the long term tracking problem by designing two complementary experts, one estimates missed detections and the other estimates false alarm, apart from this, a re-detection scheme is designed to achieve long term tracking. In [15], a sparsity-based collaboration of discriminative and generative modules are proposed. Hong et al. [54] adopt the hierarchical appearance model to track object through multi-level. In MEEM tracker [55], multiple experts are used to handle the model drift problem, which shows high tracking efficiency.

The proposed method adopts correlation filter as the base tracker. Here, we review the correlation filter methods briefly. Recently, correlation filter based tracking methods have attracted great attention due to its high efficiency [31, 32, 9, 33, 34, 35]. Since Bolme et al. [31] propose a minimum output sum of squared error filter for tracking, correlation filter began to re-attract attention as a commonly used method in signal processing. After that, Henriques et al. [32] propose to use circular image patches as dense samples to train the correlation filter in kernel space with low computation load. The above methods are based on gray-level feature. The work is further extended to HOG feature in the KCF tracking algorithm [9]. In [33], color attributes are added to the framework of [32], and an adaptive dimension reduction technique is proposed, which demonstrates the importance of color feature in object tracking. Other extended work, such as in [34, 112, 91, 35], the scale variation, long term tracking, even long short term memory scheme are added to the correlation filter tracker. In [34], the accurate scale estimation is obtained by treating translation and scale correlation separately, and a one-dimensional scale correlation filter is used to measure scale change.

Our work is most close to MEEM [55], but with significant differences summarized as follows:

- In [55], the online SVM method is adopted as the base tracker, and the grid searching method is used to sample image patches. Our method introduces the ridge regression model to learn the temporal context correlation of the object rather than the binary classifier (online SVM).
- In [55], multiple experts are regarded independently, and the entropy based loss function is defined on the single expert. Furthermore, since our base tracker of correlation filter uses the regression model with dense sampling scheme, the response map shows much less ambiguity than the binary classifier. Therefore, only the entropy based loss function will not work in our method. Since the drift tracker tends to be more confident to its own estimation, in this chapter, we further pay more attention to the collaborative efforts of multi-expert rather than the single one, and propose to define the expert score in the semi-supervised learning manner, which describes the consistency and ambiguity of the multi-expert in a unified formulation.
- We additionally take target scale variation into consideration which [55] cannot deal with.

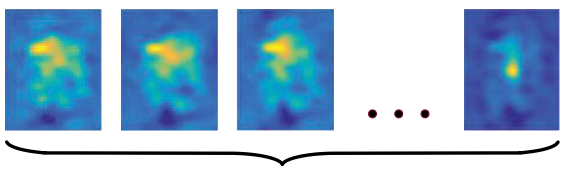
## 4.3 The Proposed Tracker

In this section, we first introduce the multi-expert ensemble framework, and then elaborate the expert selection criteria.

### 4.3.1 Tracker Snapshot Based Multi-Expert Tracking Framework

As illustrated in Fig. 4.2, given a base tracker which updates every frame, let $\mathcal{T}_t$ denotes the tracker snapshot (expert) trained up to time $t$ (In the following, we do not differentiate tracker snapshot from expert). Until time $T$, we have an expert ensemble $\mathbf{E} := \{\mathcal{T}_{t_1}, \mathcal{T}_{t_2}, ..., \mathcal{T}_T\}$, where $\mathcal{T}_T$ represents the tracker at the current time. At each time step, a score is calculated and assigned to each expert in the ensemble, the best expert is determined by its accumulative score within a pre-defined temporal window:

$$\mathcal{T}^* = arg \max_{\mathcal{T} \in \mathbf{E}} \sum_{t \in [T-\Delta, T]} \mathcal{S}_{\mathcal{T}}^t, \tag{4.1}$$



$$\mathbf{E} := \{ \quad \mathcal{T}_{t_1} \quad \mathcal{T}_{t_2} \quad \mathcal{T}_{t_3} \quad \cdots \quad \mathcal{T}_{T} \quad \}$$

Fig. 4.2 Tracking snapshot based multi-expert framework illustration.

where $\mathcal{S}_{\mathcal{T}}^{t}$ is the score of expert $\mathcal{T}$ at time $t$, and $\Delta$ is the temporal window size.

### 4.3.2 Multi-Expert Selection Criteria

It is very important to define the expert score. As each expert, especially the expert that is prone to drift, tends to be more confident to their own predictions, the expert score cannot be defined simply based on the likelihood given by the response value. Inspired by [55], the expert score is formulated by the label instead of the response values of the experts. Specifically, the expert ensemble proposes several target candidates each time. For each instance in the target candidates, the label is uncertain (be positive or negative), which can be regarded as the semi-supervised partial label learning problem [113, 114]. In the partial label learning problem, training data set is denoted by $\mathcal{D} = \{\mathbf{x}_i, \mathbf{z}_i\}$, where $\mathbf{x}_i$ is the data sample, and $\mathbf{z}_i$ indicates the possible label set that contains the true label of $\mathbf{x}_i$.

We aim to define the expert score to describe the consistency and the ambiguity of the expert relative to the others simultaneously. Motivated by [55, 114], the semi-supervised learning problem is solved in a MAP framework that maximizes the log posterior probability of the model parameterized by $\theta$:

$$\mathcal{P}(\theta, \eta; \mathcal{D}) = \mathcal{L}(\theta; \mathcal{D}) - \eta H_{emp}(Y|X, Z; \mathcal{D}, \theta), \tag{4.2}$$

where $\mathcal{L}(\theta; \mathcal{D})$ is the log likelihood of the model parameterized by $\theta$, and $H_{emp}(Y|X, Z; \mathcal{D}, \theta)$, similar to that in [55], is the empirical conditional entropy prior conditioned on the training data $X$ and their possible label set $Z$. The scalar $\eta$ is the regularization coefficient to control the trade-off between the likelihood and the prior.

In [55], the score definition only adopts the entropy to favor the expert with low ambiguity. The entropy score is suitable for the base tracker whose response map is relatively ambiguous, for instance, the grid searching methods or the discriminative methods with binary classifier. However, we find in the experiment that when using the dense searching methods or the regression base trackers, only adopting the entropy score does not work well. Therefore, in the following, we use Eq. 4.2 to define the expert score, where the log likelihood term describes the consistency of the expert relative to the others, and the entropy prior represents the expert ambiguity to the target candidates provided by the expert ensemble.

At each time step, the expert ensemble $\mathbf{E}$ proposes a set of possible target candidate set $X = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$, which are the image patches corresponding to all the local maxima given by the multi-expert. The label of the sample in the data set is denoted by $y_i = (l_i, \omega_i)$, where $l_i \in \mathbb{R}^2$ is the coordinate location of sample $\mathbf{x}_i$ and $\omega_i \in \{1,0\}$ indicates whether the sample belongs to the foreground or background. Thus the label $Y = \{y_1, y_2, ..., y_n\}$ of the candidate set $X$ resides in a very high dimension $\mathcal{Y} \in (\mathbb{R}^2 \times \{1,0\})^n$. Now we adapt Eq. 4.2 by applying it to data $X$ and the possible label set $Z$:

$$\mathcal{P}(X, Z, \theta_{\mathcal{T}}) = \mathcal{L}(\theta_{\mathcal{T}}; X, Z) - \eta H(Y|X, Z; \theta_{\mathcal{T}}). \tag{4.3}$$

For the likelihood term $\mathcal{L}(\theta_{\mathcal{T}}; X, Z)$, we further extract the global maxima $X_g = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{|\mathbf{E}|}\}$ of the multi-expert from $X$ since these samples are more representative for describing the relationship of the experts. In semi-supervised learning paradigms, graph-based method is one class of typical ways to model the relationship of the data, where each node represents a sample in the data set and the edge denotes the strength proportional to the similarity of the pair-wise data [115]. In the graph-based method, the edges can be described by the affinity matrix $W$. In our problem, the affinity matrix describes the labels instead of data, which is based on the Gaussian function:

$$W_{mn} = \begin{cases} \exp(-\frac{\|l_m - l_n\|^2}{2\sigma^2}), & \text{if } m \neq n \\ 0, & otherwise, \end{cases} \tag{4.4}$$

where $m, n \in \{1, 2, ..., |\mathbf{E}|\}$, and $\sigma > 0$ is the parameter specified for bandwidth of the Gaussian function. Here, we omit the label element $\omega$ since $X_g$ is the global maxima set for the experts. Intuitively, if the labels of the data are more similar, the likelihood should be higher. Moreover, in order to differentiate the expert that is more consistent with the others from its counterparts, we define the likelihood as: $\mathcal{C}(\theta_{\mathcal{T}}; X, Z) = \sum_{n=1}^{|\mathbf{E}|} W_{mn}$. Afterward, the log likelihood in Eq. 4.3 is obtained by calculating the natural logarithm of $\mathcal{C}(\theta_{\mathcal{T}}; X, Z)$.

In actual tracking scenario, the expert tends to be ambiguous due to continuous target appearance variations, especially when there are background clutters, heavy occlusions and abrupt motions. The entropy prior in Eq. 4.3 is used to give penalty to the expert ambiguity. As the entropy is computed in a very large label space $\mathcal{Y}$, a space narrowing strategy should be applied for efficiency. Here, we use the entropy term similar to [55], but with different space narrowing scheme that will be detailed in Sec. 4.4.

To be more specific, the nearby samples in the candidate set $X$ are merged together to get a new candidate set $X_H$, so that the samples in $X_H$ are sparsely distributed and do not heavily overlap with each other. Therefore, it is assumed only one of the samples in $X_H$ is the true target. Then the possible label set is constrained within $Z = \{Y^1, Y^2, ..., Y^{|X_H|}\}$, where $Y^k = ((l_1^k, \omega_1^k), (l_2^k, \omega_2^k), ..., (l_{|X_H|}^k, \omega_{|X_H|}^k)$, and $\omega_i^k$ is the positive label only when $k = i$, meaning that only one sample in $X_H$ can be the true target. After the above step, the entropy can be evaluated as:

$$H(Y|X,Z;\theta_{\mathcal{T}}) = -\sum_{Y\in\mathcal{Y}} P(Y|X,Z,\theta_{\mathcal{T}}) \log P(Y|X,Z,\theta_{\mathcal{T}}). \tag{4.5}$$

The probability in Eq. 4.5 is obtained by:

$$P(Y|X,Z,\theta_{\mathcal{T}}) = \frac{\delta_Z(Y)P(Y|X;\theta_{\mathcal{T}})}{\sum_{\mathbf{y}\in\mathcal{Y}} \delta_Z(\mathbf{y})P(\mathbf{y}|X;\theta_{\mathcal{T}})}, \tag{4.6}$$

where

$$\delta_Z(Y) = \begin{cases} 1, & \text{if } Y \in Z \\ 0, & \text{otherwise.} \end{cases} \tag{4.7}$$

Since the target candidates do not substantially overlap with each other, we assume the expert decisions on them are independent. Therefore, the probability in Eq. 4.6 is calculated by:

$$\begin{aligned} P(Y|X;\theta_{\mathcal{T}}) &= \prod_i P(l_i, \omega_i|\mathbf{x}_i;\theta_{\mathcal{T}}) \\ &= \prod_i P(l_i|\omega_i)P(\omega_i|\mathbf{x}_i;\theta_{\mathcal{T}}), \end{aligned} \tag{4.8}$$

where $P(l_i|\omega_i)$ is calculated by the motion model that will be detailed in Sec. 4.4.

### 4.3.3 Base tracker of Regression Correlation Filter

In recent years, correlation filter based trackers have gained much attention due to its high efficiency and robustness. Correlation trackers have show their outstanding performance in public evaluation dataset and benchmark [116, 2]. For the purposes of accuracy and low computational costs, we train the correlation filter as the base tracker in the dual space. It is worth noting that by deriving the correlation filter in the dual space, multi-channel correlation filter can be computed by only element-wise operations, which avoids the matrix inversions as in [37, 117]. We use the ridge regression model to learn the correlation of the temporal target context [9, 32]. In addition, by taking all the circular shifts of image patch into consideration, the model presents less ambiguous response map than the binary classifier, which is more favorable for the likelihood in Eq. 4.3 to take effect.

Correlation filter tracker models the appearance of the target on an extended $M \times N$ image patch $\mathbf{x}$ which is centered by the object position. The goal is to find a classifier $f(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle$ to make prediction for the probability of image patch. Instead of sampling image patch with steps, the classifier is trained with all the 2D circular shifts of $\mathbf{x}_i$, where $i \in \{0,...,M-1\} \times \{0,...,N-1\}$. Each training example $\mathbf{x}_i$ is assigned a training label generated by Gaussian function $y_i$. The training goal is to minimize the regression error:

$$\min_{\mathbf{w}} \sum_i (\langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle - y_i)^2 + \lambda \|\mathbf{w}\|^2, \tag{4.9}$$

where $\phi$ is the mapping to the kernel space, and $\lambda$ is the regularization parameter that controls overfitting. With kernel trick, $\mathbf{w}$ can be denoted by a linear combination of the training samples: $\mathbf{w} = \sum_i \alpha_i \phi(\mathbf{x}_i)$, where $\alpha$ is the dual space coefficients of $\mathbf{w}$. Given the kernel defined by $\kappa(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle$, the classifier is derived by $f(\mathbf{x}) = \sum_i \alpha_i \kappa(\mathbf{x}_i, \mathbf{x})$. Then the optimization problem is transformed under $\alpha$ instead of $\mathbf{w}$. Let the hat symbol "^" denotes the Discrete Fourier Transform (DFT). According to [9], for a unitarily invariant kernel, $\alpha$ is derived as:

$$\hat{\alpha} = \frac{\hat{\mathbf{y}}}{\hat{\mathbf{d}}_{\mathbf{xx}} + \lambda}, {}^{1} \tag{4.10}$$

where $\hat{\mathbf{d}}_{\mathbf{xx}}$ is the so-called kernel correlation whose i-th element is $\kappa(\mathbf{x}_i, \mathbf{x})$. The kernel correlation can also be computed efficiently in the Fourier domain. Particularly, for linear kernel, when the input patch $\mathbf{x}$ has multiple channels, which is concatenated by individual matrices of $C$ channels, i.e. $\mathbf{x} = [\mathbf{x}_1, ..., \mathbf{x}_C]$, the kernel correlation can be computed by:

$$\mathbf{d}_{\mathbf{xx}'} = \mathcal{F}^{-1}(\sum_c \hat{\mathbf{x}}_c^* \odot \hat{\mathbf{x}}_c'), \tag{4.11}$$

where $\mathcal{F}^{-1}$ is the Inverse DFT (IDFT), and $\odot$ denotes the element-wise product. After the above training procedure, the detection task is carried out on an image patch $\mathbf{z}$ in the new frame within the $M \times N$ window, which is centered at the last target position. The response map can be evaluated by:

$$f(\mathbf{z}) = \mathcal{F}^{-1}(\hat{\mathbf{d}}_{\mathbf{xz}} \odot \hat{\alpha}). \tag{4.12}$$

[1]For convenient, in [9], all the equations are derived using circulant matrix [118] formed by 1D vector. Actually, in 2D signals, for purpose of obtaining Eq. 4.10, it is more convenient to derive the equation by arranging the 2D data into 1D vector through *lexicographic ordering*, whose corresponding formed matrix is block circulant with circulant block (BCCB) [119, 120]. BCCB matrix has similar properties with circulant matrix. In addition, let $l(\mathbf{x})$ be the lexicographic ordering of 2D $M \times N$ image patch $\mathbf{x}$, and $\hat{l(\mathbf{x})}$ be the 2D DFT of $l(\mathbf{x})$. To keep $\hat{l(\mathbf{x})}$ a vector, the 2D DFT operation on vector $l(\mathbf{x})$ can be represented by $\hat{l(\mathbf{x})} = (\mathbf{F}_M \otimes \mathbf{F}_N) l(\mathbf{x})$, where $\mathbf{F}_M$ and $\mathbf{F}_N$ are the DFT matrix for $M$ and $N$ length 1D signals respectively, and $\otimes$ is the Kronecker product [119]. The same representation can be derived for IDFT. However, omitting the derivation procedures, all the conclusions of Eq. 4.10 $\sim$ 4.12 can be computed directly in 2D without *lexicographic ordering*.

Therefore, the new position of the target is determined by the maximum of $f(\mathbf{z})$. To move discontinuities at the image boundaries of the response map, the input feature channels are weighted by a cosine window [31]. To adapt to the appearance change of the target, the linear interpolation strategy is conducted on $\alpha$ and $\mathbf{x}$:

$$\hat{\alpha}^t = (1-\gamma)\hat{\alpha}^{t-1} + \gamma\hat{\alpha}, \tag{4.13}$$

$$\hat{\mathbf{x}}^t = (1-\gamma)\hat{\mathbf{x}}^{t-1} + \gamma\hat{\mathbf{x}}, \tag{4.14}$$

where $\gamma$ is the learning rate.

In the correlation filter base tracker, we employ the PCA-HOG feature described in [92]. Besides the HOG feature which puts more emphasis on the object shape, we further add the color feature to promote the tracker performance. Here we apply color attribute feature to map the RGB values to the probabilistic 11 dimensional color representation [94, 33].

In order to deal with the scale variation of the target, we adopt a scale adaptive method to our tracker. Unlike [34, 35], our method estimates the translation and target scale simultaneously. Let $M_t \times N_t$ denotes the search window size at time $t$, we first establish a target pyramid through cropping image patches, all of which are centered at the target position of time $t-1$, each of size $(1+as)M_t \times (1+as)N_t$, where $a$ is a constant scalar of the scale factor, and $s \in \{\lfloor -\frac{N_s-1}{2} \rfloor, \lfloor -\frac{N_s-3}{2} \rfloor, ..., \lfloor \frac{N_s-1}{2} \rfloor\}$ is the scale index. Then all the $N_s$ image patches are resized to the target template size. After that, the response map of each cropped image patch can be evaluated according to Eq. 4.12, all of which constitute the response pyramid. Finally, the accurate scale index is indicated by the maximum of the response pyramid, as well as the translation (which should multiply by its ratio relative to the template size).

### 4.3.4 Scale-adaptive Multi-Expert (SME) Tracker

Given the above expert selection criteria and the base tracker of correlation filter, we propose our tracker, named Scale-adaptive Multi-Expert (SME).

The snapshots in the expert ensemble are stored chronologically at intervals of $\Delta t$ frames. When the number of experts exceeds the maximum number $N_{\mathbf{E}}$, the oldest expert is discarded. At each frame, each expert in the ensemble gets its own decision of the target position by calculating the maximum value of the expert response map. In addition, the expert ensemble proposes the potential target candidates $X$. After that, expert scores are calculated each frame by Eq. 4.3. Whenever there is a disagreement among the experts, the best expert is selected according to their accumulative score define by Eq. 4.1, and displaces the current expert to be the current tracker. Otherwise, the target is tracked by the current expert $\mathcal{T}_T$. Note that only expert $\mathcal{T}_T$ is updated, so the whole algorithm has low computation.

At the same time, target scale is estimated according to the method described in Sec. 4.3.3. Generally, scale variation is much smaller than that of translation. For computation efficiency, the scale estimation is only conducted on the current expert $\mathcal{T}_T$. We find this strategy very effective in practice.

## 4.4 Implementation

The whole algorithm procedures of our SME tracker is shown in the following Algorithm flowchart.

**Algorithm 3:** SME Tracker

```
input : Initial target bounding box $\mathbf{x}_1$
output: The estimated target state $\mathbf{x}_t = (\hat{x}_t, \hat{y}_t, \hat{s}_t)$
$\mathbf{E} \leftarrow \mathcal{T}_1$
repeat
    Get the target candidate set $X$ by $\mathbf{E}$;
    for $\mathcal{T} \in E$ do
        if $\mathcal{T} = \mathcal{T}_t$ then
            Build the target pyramid at $(\hat{x}_{t-1}, \hat{y}_{t-1})$;
            Get the response pyramid, estimate the target position $(x_{\mathcal{T}_t}.y_{\mathcal{T}_t})$ and scale $\hat{s}_t$;
        else
            Get the response map and estimate the target position $(x_{\mathcal{T}}, y_{\mathcal{T}})$;
        Compute the expert score $\mathcal{S}^t_{\mathcal{T}}$ according to Eq. 4.3;
    if expert disagreement is reported then
        Select $\mathcal{T}^* \in \mathbf{E}$ according to Eq. 4.1;
        $\mathbf{x}_t = (x_{\mathcal{T}^*_t}, y_{\mathcal{T}^*_t}, \hat{s}_t)$;
        $\mathcal{T}_t \leftarrow \mathcal{T}^*$;
    else
        $\mathbf{x}_t = (x_{\mathcal{T}_t}, y_{\mathcal{T}_t}, \hat{s}_t)$;
    if mod $(t, \Delta t) == 0$ then
        $\mathbf{E} \leftarrow \mathbf{E} \cup \mathcal{T}_t$;
        discard the oldest snapshot when $|\mathbf{E}| > N_{\mathbf{E}}$;
    Update $\mathcal{T}_t$;
until Last frame of video sequences;
```

Some implementation details are discussed below.

The target positions decided by each expert are processed by hierarchical clustering according to their spatial distribution. If the clustering result has more than one class, a disagreement is reported. In order to obtain the target candidate set $X$, we first use the non-maxima suppression method to get the local maxima of the multi-expert response maps, and the target candidates are the image patches corresponding to the local maxima whose response value is greater than the pre-defined threshold $\varepsilon$. The samples are then processed by hierarchical clustering according to their spatial positions. Samples of the same cluster are merged at their mean positions to avoid heavily overlap. The sample probability $P(\omega_i|\mathbf{x}_i;\theta_{\mathcal{T}})$ in Eq. 4.8 for $\omega_i = 1$ is naturally obtained by the response map of the base tracker. The corresponding motion model $P(l_i|\omega_i)$ for foreground, i.e. $\omega_i = 1$ is set as the Gaussian distribution centered at the previous target location. On the other hand, the probability for $\omega_i = 0$ is got by $1 - P(+1|\mathbf{x}_i;\theta_{\mathcal{T}})$, and the motion model for background $\omega_i = 0$ is set as the uniform distribution.

The parameters setup of the multi-expert framework is as follows. The max number of experts $N_{\mathbf{E}}$ is set to 4. We find the performance just falls slightly with the expert number increases larger than 4, however, with higher computation load. The frame interval $\Delta t$ is set to 50, and we find the performance is not sensitive to tens of frames such as 30 or 40. Let the template target size denoted by $s$. We set the cutoff distance of the hierarchical clustering by $s/2$. The parameter $\sigma$ for computing the likelihood in Eq. 4.4 is set to $s/3$ according to the 3-sigma rule of Gaussian function. Note that all the response maps of the expert ensemble (including each layer of response pyramid generated by the current expert) are of the same template size. Therefore the above parameters are not influenced by target size. The trade-off parameter $\eta$ in Eq. 4.3 is set to 15. The candidate selection threshold $\varepsilon = 0.8$.

For the correlation filter base tracker, only linear kernel is applied, and the padding size and interpolation learning rate are set to 1.8 and 0.01 respectively. The number of target pyramid layers $N_s$ is 9, and the scale factor $a$ is 0.005. The template size is set to the initial target size.

## 4.5 Experiments

In this section, we evaluate our SME tracker on two large dataset, one is the 50 sequences Object Tracking Benchmark (OTB-50) [2] used in the last chapter, the other is the 60 sequences VOT2015 Challenge dataset [11]. On the former dataset, we compare the proposed tracker with state-of-the-art trackers to demonstrate its excellent performance. Then, the tracker is tested on sequences of eight main attributes to analysis the performance of SME in different scenarios. We also decompose SME into different parts to analysis the effectiveness of the proposed framework. To further verify the efficiency of our tracker, we test SME on the new VOT2015 dataset, which contains 60 sequences, the experimental result of the VOT dataset is reported for evaluation.

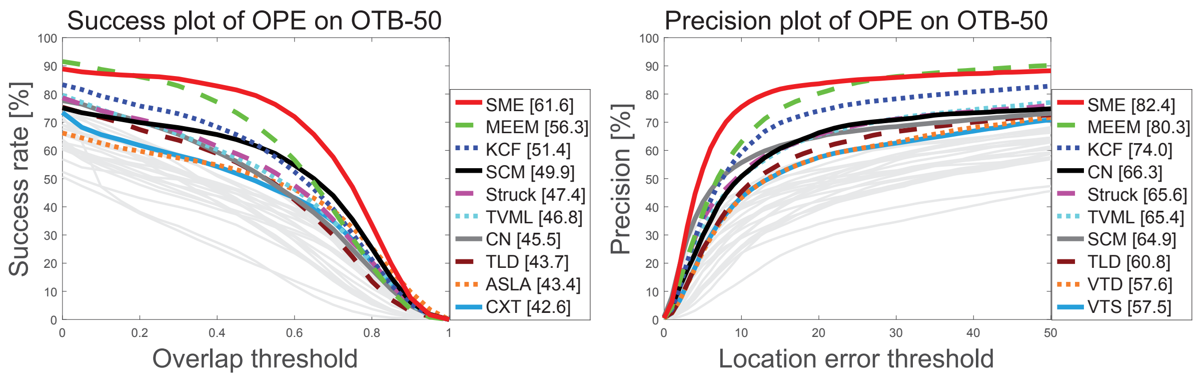


Fig. 4.3 The precision plot and success plot over 50 sequences benchmark using one pass evaluation (OPE). The legend illustrates the score of the threshold 20 for the precision plot, and the area under curve (AUC) for the success plot. Only top-10 trackers are colored, and the others are shown in gray curves. Legend of the same color denotes the same rank.

### 4.5.1 Experimental Setup and Metrics

The proposed SME tracker is implemented in Matlab&C. Although with multiple experts, our tracker runs $>$ 20fps on a 3.20GHz CPU PC, mostly due to the efficiency of the correlation filter. The parameters setup is in accordance to the description in Sec. 4.4.

The Metrics for OTB-50 has been introduced in the last chapter.

The VOT2015 dataset contains 60 short challenging sequences. The sequences are annotated using rotated bounding box in order to provide highly accurate ground truth, which is different from OTB-50 annotated by rectangles. The VOT2015 dataset is evaluated by two criteria: (i) accuracy and (ii) robustness. The accuracy measures the overlap between the tracking result and ground truth. The robustness measures how many times the tracker loses the target. The details for the evaluation method can be referred to [11, 121].

### 4.5.2 Evaluation on Object Tracking Benchmark-50

**Overall Performance.**

Besides the trackers provided by [2], we add 4 excellent trackers including KCF [9], CN [33] and MEEM [55], as well as the proposed TMVL tracker in the last chapter. According to the evaluation methods by OTB-50, the one-pass evaluation (OPE) performance is illustrated in the Success Plot and Precision Plot shown in Fig. 4.3.

As the same illustration in the last chapter, we plot the top-10 among all the compared trackers. As shown in the plots, our tracker achieves 61.6% success score and 82.4% precision score, both of which rank first among all the trackers. Compared with the TVML tracker proposed in the last chapter, SME has improved the success and precision performance a lot by

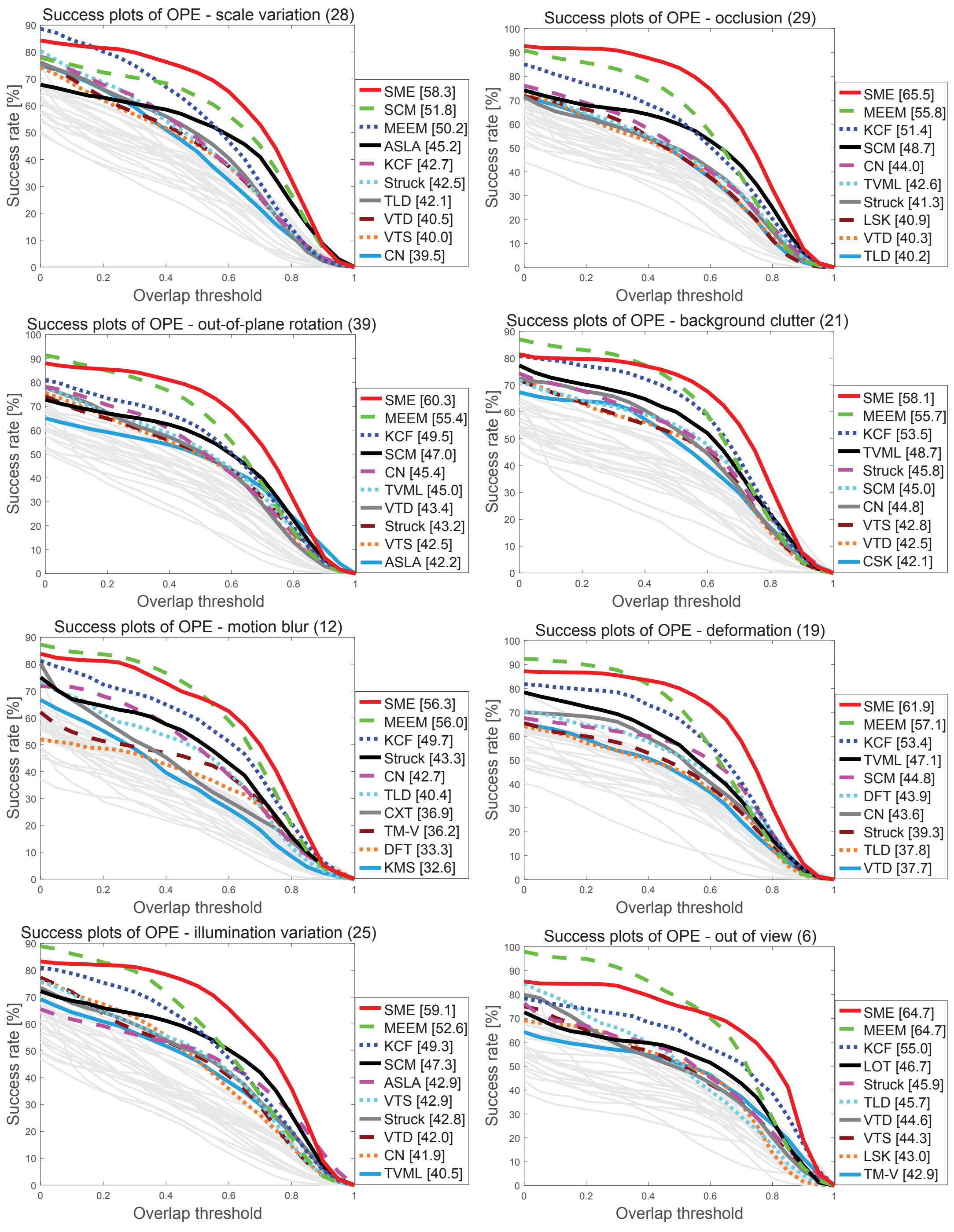


Fig. 4.4 The success plots of eight attributes of the benchmark, i.e. scale variation, occlusion, out-of-plane rotation, background clutter, motion blur, deformation, illumination variation and out of view. The legend illustrates the AUC score for each tracker.

Table 4.1 Success and Precision Score of the SME Analysis

| | SME | SME-base | SME-sfix |
|---|---|---|---|
| **Success [%]** | 61.6 | 59.5 | 58.3 |
| **Precision [%]** | 82.4 | 78.2 | 81.0 |

31.6% and 26.0% respectively. Particularly, MEEM is most similar to our tracker. Compared to MEEM, SME surpasses it with large margin, especially exceeds in the success score by 9.4%. KCF is the also the correlation filter based method, which can be regard as our base tracker. Compared to KCF, our tracker improves the overlap success and precision score by 19.8% and 11.4% respectively. The overall plots demonstrate our tracker is effective and promising.

### Attribute-Based Performance

In this experiment, the benchmark sequences are divided into 8 main attributes to evaluate the tracker in different scenarios. As SME is capable of estimating the target scale, the AUC score of success plot here can measure the tracker performance more accurate than precision plot of one threshold. Therefore, we report the eight main attributes of success plots in Fig. 4.4.

Fig. 4.4 shows the success plots of eight attributes of the benchmark, i.e. scale variation, occlusion, out-of-plane rotation, background clutter, motion blur, deformation, illumination variation and out of view. As illustrated in the plots, SME ranks first in all the attributes (the same score as MEEM in out-of-view attribute). In addition, SME achieves higher score than TVML. SCM shows high score of 51.8% in the scale variation attribute, while SME performs better with 58.3% success scores. The MEEM ranks second in all the plots, which demonstrate that ensemble tracking method is more preferable in complex tracking scenarios. Particularly, in the attributes of scale variation, occlusion, out-of-plane rotation, background clutter, deformation and illumination variation, SME exceeds the second rank tracker by 12.5%, 17.4%, 8.8%, 4.3%, 8.4% and 12.3% respectively. Among all the attributes, the scale variation performance is improved significantly, which shows our scale scheme is very effective. In addition, SME also gets more favorable scores than other correlation filter based trackers, KCF [9] and CN [33], which demonstrates the effectiveness of the multi-expert framework.

### Component Analysis

To further demonstrate the effectiveness of the proposed tracker, we decompose our approach into two trackers with part of the features of the original SME: (i) SME-base, the base correlation

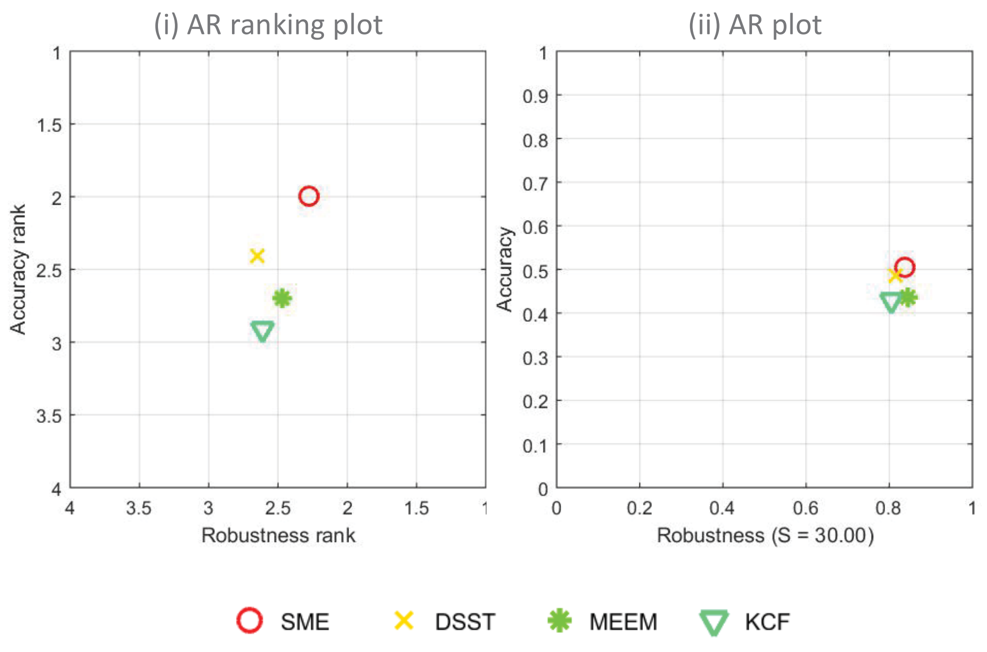


Fig. 4.5 The AR ranking plot and the AR plot for VOT2015 Challenge Dataset. The tracker is better if its legend resides closer to the top-right corner of the plot. $S$ is the data visualization parameter.

tracker. (ii) SME-sfix, the original SME without scale estimation. We summarize their success and precision score in Table. 4.1.

From the table, both the success and precision score of SME are higher than its counterparts. Moreover, the precision score of SME-base decreases from 82.4% to 78.2% compared to original SME, which means that the multi-experts framework is important in improving the CLE score in tracking. On the other hand, when removing the scale estimation, the success score of SME-sfix declines immediately. However, the precision of SME-sfix falls relatively smaller compared to SME-base. It is reasonable since the multi-expert framework pays more attention to correct the estimation error of target translation by selecting reliable historical tracker snapshots, while the scale estimation is more related to overlap metric. Therefore, the combination of the two gives satisfactory effect.

### 4.5.3 Evaluation on VOT2015 Challenge Dataset

The number of sequences in VOT2015 Challenge Dataset has been enlarged to 60 compared to VOT2013 [122] and VOT2014 [116], whose numbers of sequences are 16 and 25 respectively. On this dataset, we compare the performance of SME with three trackers, DSST [34], MEEM [55] and KCF [9]. MEEM and KCF are perform well on OTB-50 dataset. Since DSST is the winner of the VOT2014 challenge [116], comparing with it can validates the performance of SME to a large extent. In addition, MEEM is close to our work, we choose to compare with it further in this larger dataset.

According to the VOT evaluation criteria [116], the overall experimental results are illustrated in two plots: (i) accuracy-robustness (AR) ranking plot and (ii) AR plot, as shown in Fig. 4.5, and the AR ranking plot is the main evaluation criteria. The AR ranking plot shows average ranking score of all the sequences for each tracker in the joint accuracy-robustness rank space. The AR plot is the data visualization shows the average accuracy-robustness data of each tracker. For both plots, the tracker is better if the legend resides closer to the top-right corner of the plots.

## 4.6 Conclusion

In this chapter, we propose an effective tracker snapshot based multi-expert tracker. The multi-expert is composed of both the current tracker and its historical snapshots. The multi-expert framework is capable of detecting the tracker drift and select the most reliable expert according to their accumulated score. As the drift expert is prone to present higher confidence to its own prediction, we extend the expert loss function based on [55] as the expert score, which can describe the consistency and ambiguity of the multi-expert in a unified formulation. Each expert is learned by the discriminative correlation filter, while the scale is estimated by searching the target pyramid. The experiments are conducted on two large tracking datasets, which demonstrate the proposed tracker performs favorably against state-of-the-art methods.

# Chapter 5

# Discrete Graph Based Multi-Expert Tracking

## 5.1 Introduction

The tracker snapshot based method proposed in the last chapter correct the tracking drift by using the tracker historical memory restoration scheme after detecting the drift. However, it still has some issues to be solved:

- The multi-expert framework is established in a heuristic form, including the temporal window size for score accumulation etc.
- Tracker drift is corrected aggressively by replacing the current tracker by the historical snapshot, which means totally discarding the recent tens and even hundreds of frames update.

The above issues motivate us to think of:

- How can we model the relationship among multiple experts instead of heuristic form ?
- Since object tracking demands high efficiency, how can we achieve the solution of the framework without sacrificing the computation load?

To solve the above issues, in this chapter, we propose a graph optimization framework to model the relationship of the multiple experts. The graph nodes are modeled by the hypotheses of the multiple experts. The proposed method is capable of automatically detecting and correcting the tracker drift by finding the optimal path with the highest score in the graph. With the efficient solver of dynamic programing, our method can implicitly analyze the trajectories and reliabilities of the multi-expert by only computing their scores in the current frame, so as to effectively correct the tracker drift with high efficiency.

The proposed framework is applied to three base trackers as special cases. We first introduce the online SVM on a budget [55, 123] as the base tracker, and show our framework can significantly improve its performance. Moreover, we adopt the regression correlation filter with hand-crafted features and CNN features respectively as the base tracker to further boost the tracking performance, which learns the temporal context correlation of the target, and an efficient scale adaptive scheme is introduced to handle the object scale changes on-the-fly. On the widely used OTB-50 benchmark [2], the proposed methods show significant improvement against other state-of-the-art methods. The proposed trackers are further tested on the new OTB-100 [124] and VOT2015 [11] dataset, which also shows its excellent performance.

In summary, in this chapter, we extend the work of multi-expert tracking framework in the last chapter. The multi-expert ensemble is still constituted by the current tracker and its past trained snapshots. To differentiate from the SME tracker in the last chapter, here, a historical snapshots multi-expert (HSME) framework is introduced to handle the tracker drift problem. We propose to use the unified discrete graph algorithm to model the multiple experts, where an exact solution exists, which can be solved efficiently by dynamic programming without sacrificing the computation load. We integrate three base trackers, online SVM and correlation filter (hand-craft feature and CNN feature) into our HSME framework, and propose the trackers named HSME-SVM, HSME-CF and HSME-deep respectively. The proposed three trackers are extensively evaluated on three big datasets, and show excellent performance against state-of-the-art methods with high efficiency.

The rest of the chapter are organized as follows. After reviewing the related work in Sec. 5.2, we introduce the proposed method in Sec. 5.3. The implementations and method analysis for the proposed trackers are detailed in Sec. 5.4. The experimental results are presented in Sec. 5.5. Finally, we conclude the paper.

## 5.2 Related Work

**Tracking-by-Detection**. Tracking-by-detection method plays a key role among numerous recent trackers. Under this framework, a discriminative classifier is trained to classify the foreground and background features [6, 8, 27, 10]. As introduced in the related work in Sec. 3.2 of Chapter 3, many tracking-by-detection methods have been proposed. For example, *SVT* tracker [6] that integrates SVM and optical flow, *MIL* tracker [8] with multiple instance learning, *CT* tracker that utilizing the idea of compressive sensing, and *Struck* tracker with structured output labels [10]. Others, such as in [125], context information is also added to promote tracking performance.

Over the last few years, correlation filter based tracking methods have been widely used due to their excellent performance and high efficiency [31, 32, 9, 33, 34]. In Sec. 4.2 of Chapter 4, we have reviewed correlation filter trackers in detail. More recently, a complementary feature integration scheme is introduced in correlation filter [126], which shows excellent performance.

Since convolutional neural networks (CNNs) have achieved great success in computer vision [1, 79, 127], there is a rising trend for introducing CNN models into visual object tracking task. In [128], a stacked autoencoder is used to learn the tracker representation. On the other hand, Hong et al. [45] adopt a learned saliency map on pre-trained CNN to predict the target state. In [41], CNN feature is integrated into the correlation filter framework, and target location is estimated hierarchically to improve the tracking robustness. Zhang et al. [129] propose a two-layer convolutional network with a lightweight structure for visual tracking without off-line training. In [130], a biologically inspired tracker is proposed based on the analysis of visual cognitive mechanism. Furthermore, in [47], a sequential online training method is proposed to overcome overfitting when fine-tuning the pre-trained deep models for visual tracking.

**Tracker Ensemble (Multi-Expert)**. As discussed in the related work section of last chapter, many tracking algorithms adopt tracker ensemble to achieve more robust tracking performance. For example, the Bayesian framework based *VTD* [13] and *VTS* [23] trackers, the bootstrapping based *TLD* [7] tracker, and the sparsity-based collaboration of discriminative and generative modules based *SCM* tracker [15]. In *MEEM* [55] and *SME* tracker proposed in the last chapter, multiple experts including historical tracker snapshots are used to handle the model drift problem, which shows high tracking efficiency. They both use a disagreement threshold to decide whether the current tracker drifts, and then select the most reliable expert according to their accumulated scores. Another multi-expert selection strategy can be found in [131]. In [131], each expert first moves forward and then backward to get pairs of trajectories, the most reliable expert is selected by analyzing their pair-wise geometry similarity.

Our work is mostly close to MEEM [55] and SME tracker of last chapter, but with significant differences summarized as follows. Firstly, in MEEM and SME, multi-expert is established in a heuristic form. With the multi-expert framework, tracker drift is first detected by the handcrafted condition and then corrected by the expert selection scheme. In our work, the above two steps are merged and modeled as the unified discrete graph optimization framework, and the tracker drift is corrected by finding the exact optimal solution. Secondly, in MEEM and SME, tracker drift is corrected aggressively by replacing the current tracker by the historical snapshot, which means totally discarding the recent tens and even hundreds of frames update. While our framework is capable of correcting the tracker drift promptly both before or after the large disagreement among the multiple experts, which is more flexible and stable. Thirdly, our method is a general framework. We test the proposed framework in three general base trackers to validate its effectiveness.

## 5.3 Proposed Method

In this section, we first introduce the discrete graph optimization model, and its solver of dynamic programming. Then we show how to model the multiple experts by the graph optimization framework, as well as how to define the graph scores. At last, we introduce the base trackers.

### 5.3.1 Relational Graph and Hypothesis Graph

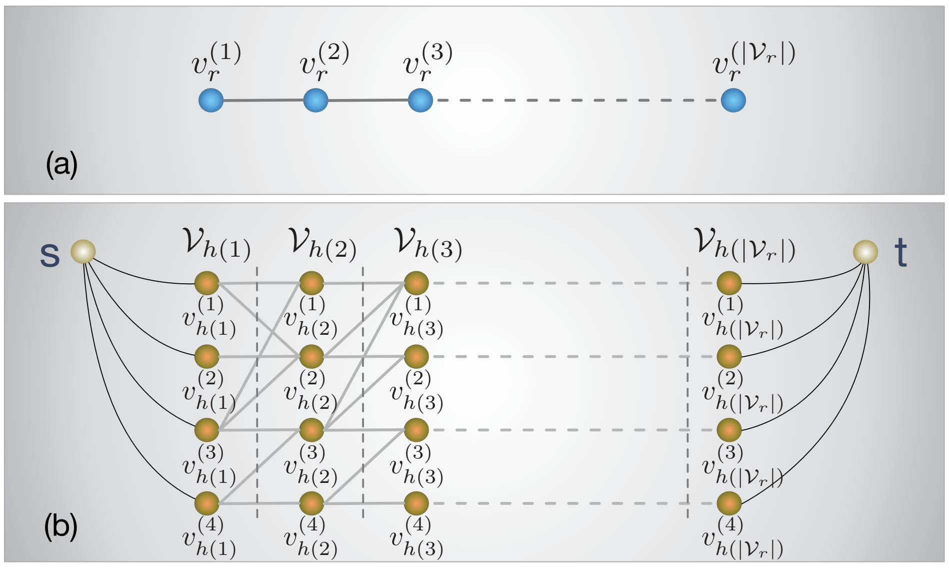
In the computer vision problem, many issues can be formulated as follows [132, 133]. Assuming there is a set of entities $V = \{v^{(n)}|_{n=1}^{N}\}$, where each entity can be in one of the following states $S = \{s^{(m)}|_{m=1}^{M}\}$, with the unary score function $\phi(s^{(m)}|v^{(n)}; v^{(n)} \in V, s^{(m)} \in S)$, which represents the likelihood that entity $v^{(n)}$ is in state $s^{(m)}$. Moreover, there is a binary compatible function $\psi(s^{(k)}|v^{(i)}, s^{(l)}|v^{(j)}; v^{(i)}, v^{(j)} \in V, s^{(k)}, s^{(l)} \in S)$, which denotes the compatibility for that entity $v^{(i)}$ in state $s^{(k)}$ and entity $v^{(j)}$ in state $s^{(l)}$. A natural objective is to assign the states for all the entities so that all of them are with highest unary scores as well as are most compatible with each other. The above issue can be regarded as a discrete optimization problem presented by the relational graph and hypothesis graph.

#### Relational Graph

A relational graph $\mathcal{G}_r = (\mathcal{V}_r, \mathcal{E}_r)$ describes the relationship of a set of entity nodes $\{v_r^{(i)}|_{i=1}^{|\mathcal{V}_r|}\}$ and relationship of pair-wise nodes represented by the edge $e \in \mathcal{E}_r$. For example, Fig. 5.1(a) shows a typical single branch relational graph.

#### Hypothesis Graph

With a relational graph $\mathcal{G}_r$, the corresponding hypothesis graph $\mathcal{G}_h = (\mathcal{V}_h, \mathcal{E}_h)$ can be established. For each entity node $v_r^{(i)} \in \mathcal{V}_r$, a set of hypothesis nodes $\mathcal{V}_{h(i)} = \{v_{h(i)}^{(k)}|_{k=1}^{|\mathcal{V}_{h(i)}|}\}$ are proposed, where $\mathcal{V}_h = \bigcup_{i=1}^{|\mathcal{V}_r|} \mathcal{V}_{h(i)}$. It means that the hypothesis nodes represent the possible states of each entity node. And the hypothesis edges can be denoted by $\mathcal{E}_h = \{(v_{h(i)}^{(k)}, v_{h(j)}^{(l)}); v_{h(i)}^{(k)} \in \mathcal{V}_{h(i)}, v_{h(j)}^{(l)} \in \mathcal{V}_{h(j)}\}$, which are built according to the structure of relational graph $\mathcal{G}_r = (\mathcal{V}_r, \mathcal{E}_r)$, where $(v_r^{(i)}, v_r^{(j)}) \in \mathcal{E}_r$. Similarly, each hypothesis node has its unary score $\phi$, and each pair-wise



$v_r^{(1)}$
$v_r^{(2)}$
$v_r^{(3)}$
$v_r^{(|\mathcal{V}_r|)}$
(a)
s
t
$\mathcal{V}_{h(1)}$
$\mathcal{V}_{h(2)}$
$\mathcal{V}_{h(3)}$
$\mathcal{V}_{h(|\mathcal{V}_r|)}$
$v_{h(1)}^{(1)}$ $v_{h(1)}^{(2)}$ $v_{h(1)}^{(3)}$ $v_{h(1)}^{(4)}$
$v_{h(2)}^{(1)}$ $v_{h(2)}^{(2)}$ $v_{h(2)}^{(3)}$ $v_{h(2)}^{(4)}$
$v_{h(3)}^{(1)}$ $v_{h(3)}^{(2)}$ $v_{h(3)}^{(3)}$ $v_{h(3)}^{(4)}$
$v_{h(|\mathcal{V}_r|)}^{(1)}$ $v_{h(|\mathcal{V}_r|)}^{(2)}$ $v_{h(|\mathcal{V}_r|)}^{(3)}$ $v_{h(|\mathcal{V}_r|)}^{(4)}$
(b)

Fig. 5.1 Graph illustration. (a) Relational Graph for single branch tree. (b) Hypothesis graph generated according to (a) assuming each entity of relational graph has four possible states. The brown balls denote the hypothesis nodes and the lines between balls represent the edges of the hypothesis graph. Note this is only the illustration and not all the edges are drawn for conciseness.

hypothesis edge has its binary compatible score $\psi$. An illustration of hypothesis graph can be found in Fig. 5.1(b).

### Discrete Graph Solver

The goal for discrete graph optimization is to select one hypothesis node for each entity node, so that to maximize the summation of their unary and binary compatible scores. Therefore, given a hypothesis graph $\mathcal{G}_h$ building upon a relational graph $\mathcal{G}_r$, the objective function for a set of hypothesis nodes $v_h = \{v_{h(i)}|_{i=1}^{|\mathcal{V}_r|}; v_{h(i)} \in \mathcal{V}_h\}$ is:

$$\mathcal{F}(v_h) = \sum_{v_{h(i)} \in \mathcal{V}_h} \phi(v_{h(i)}) + \beta \sum_{(v_{h(i)}, v_{h(j)}) \in \mathcal{E}_h} \psi(v_{h(i)}, v_{h(j)}), \tag{5.1}$$

where $\beta$ is a parameter to trade off the unary and binary weights. The goal is to find an optimal set of nodes to maximize the above objective function, i.e. $v_h^* = \arg\max_{v_h}(\mathcal{F}(v_h))$.

This general problem of discrete optimization is NP-hard, however, if the entities of the relational graph are connected in a tree structure, the problem can be solved efficiently by dynamic programming in polynomial time [132]. Furthermore, in this chapter, the proposed problem can be abstracted as a degenerate tree structure (single branch tree as shown in Fig. 5.1).

Now we show how to use dynamic programming to get the solution $v_h^* = \{v_{h(i)}^*|_{i=1}^{|\mathcal{V}_r|}; v_{h(i)}^* \in \mathcal{V}_h\}$ that maximizing the objective function of Eq. 5.1. Given a relational graph of single branch tree, and let $\mathcal{S}_i(k)$ denotes the maximum accumulated unary and binary score of hypothesis node $v_{h(i)}^{(k)}$ for the first $i$th entity, Eq. 5.1 can be solved by dynamic programming through the recursive equations:

$$\mathcal{S}_i(k) = \phi(v_{h(i)}^{(k)}) + \max_l(\mathcal{S}_{i-1}(l) + \beta\psi(v_{h(i-1)}^{(l)}, v_{h(i)}^{(k)})). \tag{5.2}$$

Once the $\mathcal{S}_i(k)$ has been computed, the optimal solution can be obtained by setting $v_{h(|\mathcal{V}_r|)}^* = \arg\max_k \mathcal{S}_{|\mathcal{V}_r|}(k)$ and tracing back by decreasing $i$:

$$v_{h(i)}^* = \arg\max_{v_{h(i)}^{(k)}}(\mathcal{S}_i(k) + \psi(v_{h(i)}^{(k)}, v_{h(i+1)}^*)). \tag{5.3}$$

The above dynamic programming algorithm runs in $\mathcal{O}(|\mathcal{V}_r|K^2)$, where $K$ is the maximum number of hypothesis nodes for each entity in $\mathcal{V}_r$.

### 5.3.2 Multi-Expert Framework modeled by Discrete Graph

#### Graph for Multiple Experts

We aim to utilize the trained tracker snapshots to build the multiple experts framework. Given a base tracker which updates every frame, we store the update record of the base tracker at intervals of certain frames as the tracker snapshot, i.e. the expert (In the following, we do not differentiate tracker snapshot from expert). Let $\mathcal{T}_t$ denotes the tracker snapshot (expert) trained up to time $t$. Until time $T$, we have an expert ensemble $\mathbf{E} := \{\mathcal{T}_{t_1}, \mathcal{T}_{t_2}, ..., \mathcal{T}_T\}$, where $\mathcal{T}_T$ represents the tracker at the current time.

Using the discrete graph introduced in Sec. 5.3.1, we model the evolution of the multi-expert as a single branch relational graph, with the entity node represents the expert ensemble. At each time step, there are $|\mathbf{E}|$ possible object state hypotheses, where each hypothesis is given by a single expert in the expert ensemble. Then the corresponding hypothesis graph is built whose node is the hypothesis given by each expert. As a result, the best hypothesis for each frame is selected according to the path that achieves the highest score from the beginning to the current frame, which can be solved by dynamic programming indicated in Eq. 5.2. An illustration can be found in Fig. 5.2, where the expert ensemble contains maximum 4 experts, and discards the oldest expert when its number exceeds 4. The notations for the graph unary score in Fig. 5.2 are the same in those of Sec. 5.3.1. In the following, we show how to define the unary and binary graph scores.

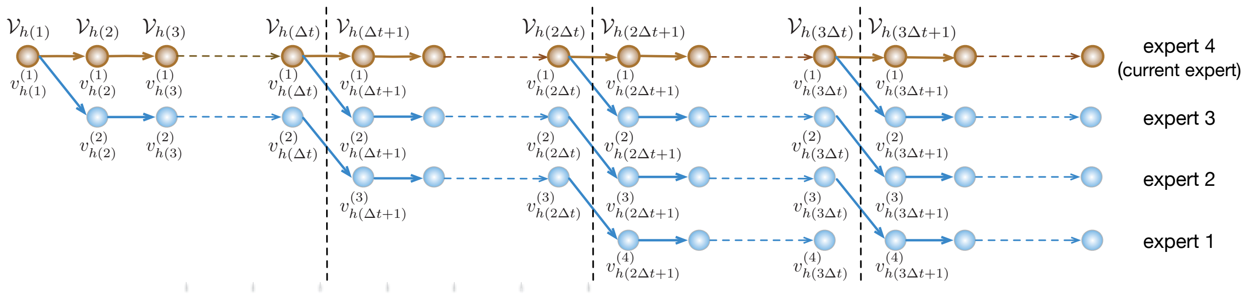


Fig. 5.2 Illustration for the multi-expert framework modeled by discrete graph. The multi-expert ensemble includes the current tracker represented by the brown nodes and the historical snapshots colored by blue nodes. The historical tracker snapshot is stored at intervals of $\Delta t$ frames, which is denoted by the black dash line, and the oldest expert is discarded if the expert number exceeds the maximum (the figure illustrates maximum 4 experts). Arrow direction denotes the same expert, and the graph edge exists in pair-wise adjacent nodes.

### Unary Graph Score and Binary Compatible Graph Score

It is very important to define the unary score and binary compatible score of the graph. As each expert, especially the expert that is prone to drift, tends to be more confident to their own predictions, the expert score cannot be defined simply based on the likelihood given by the response value. We use the expert score defined in Eq. 4.3 of Chapter 4 which formulated by the label instead of the response values of the experts.

According to the motivation of proposed framework, the design of binary compatible score should satisfy two criteria: a) Let the multi-expert framework favors the current tracker when the object is tracked stably, i.e. it's unnecessary to switch expert frequently under this situation; b) The same expert is regarded as compatible in order to get the accumulative score of the same expert. Therefore, according to criterion a), we use the Gaussian function to define the binary compatible score for node $m$ and node $n$ based on the pair-wise label similarity:

$$\rho_{mn} = \exp(-\frac{\max\{0, \|l_m - l_n\| - r\}^2}{\tau^2}), \tag{5.4}$$

where $l_m$ and $l_n$ are the positions given by the global maxima of expert m and expert n respectively, and $r$ is the threshold that has the same effect of "hinge loss", which means historical trackers are regarded as compatible with the current tracker if their label similarity is less than $r$. To achieve criterion b), for the same expert, we set the binary score as $\max(\rho_{mn})$.

### 5.3.3 Base Trackers for Multi-Expert Framework

As described in Sec. 5.3.2, the graph score is based on the target candidate set $X$ and its possible labels, therefore the proposed framework is compatible to most of the base trackers that are able to provide the response map for the target, such as the grid searching based tracking-by-detection methods or the dense searching based correlation filter methods. Due to the high efficiency and the impressive performance of SVM trackers [55, 10] and correlation filter trackers [9, 34], we choose the online SVM on a budget algorithm and the regression correlation filter (CF) as special cases for the proposed framework respectively. Here, we propose three Historical Snapshots Multi-Expert (HSME) trackers, HSME-SVM for online linear SVM, and HSME-CF, HSME-deep for regression correlation filter with hand-crafted features and CNN features respectively.

#### Base tracker of Online Linear SVM

The online SVM base tracker is inspired by the online SVM training strategy in [55, 123], which adopt the compact prototype sets to make an approximation to offline SVM. Furthermore, a prototype merge scheme is introduced to keep a budget of the model size.

A SVM base tracker $\mathcal{T}$ contains a prototype set $\mathcal{P} = \{\zeta_i = (f(\mathbf{x}_i), \omega_i, v_i)\}_{i=1}^{B}$, where $f(\mathbf{x}_i)$ is the feature vector of image patch $\mathbf{x}_i$, $\omega_i \in \{1, 0\}$ is the binary class label and $v_i$ represents the number of support vectors indicated by instance $\zeta_i$. The online training strategy is operated on both the prototype set and the new data set $\mathcal{D} = \{\mathbf{x}_j, y_j\}_{j=1}^{J}$ by minimizing the following objective function:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 + C\{\sum_{i=1}^{B} \frac{v_i}{N_{\omega_i}} L(\omega_i, \mathbf{x}_i; \mathbf{w}) + \sum_{j=1}^{J} \frac{1}{N_{y_j}} L(y_j, \mathbf{x}_j; \mathbf{w})\}, \tag{5.5}$$

where

$$N_1 = \sum_{\omega_i=1} v_i + \sum_{y_j=1} 1, \quad N_0 = \sum_{\omega_i=0} v_i + \sum_{y_j=0} 1. \tag{5.6}$$

The loss function defined in Eq. 5.5 lets the prototype with larger support vector numbers exert more influence on SVM training. When the prototype set size is larger than a pre-defined budget number, pair of prototype instances of the same label are merged into a single instance. To tackle the effect that negative samples have much larger density than that of positive ones, positive prototype instances are not merged until the instance number reaches a predefined bound.

The feature of image patch is extracted in the CIE Lab color space, while a local rank transform is applied on L channel to obtain the illumination invariance. Then the single rank image and the three channel CIE Lab image are integrated into a four channel image. More details can be referred to [123, 55].

#### Base tracker of Regression Correlation Filter

The base tracker of hand-craft feature correlation filter has been introduced in Sec. 4.3.3 of Chapter 4. Here, by using the same base tracker as in Chapter 4, we integrate it into the proposed discrete graph tracking framework, and propose HSME-CF.

#### Base tracker of CNN feature

For HSME-deep tracker, we introduce the pre-trained CNN feature into the regression correlation filter. Unlike [47, 45], which fine-tune the CNN models online, here, we only use the pre-trained model of VGG-Net with 19 layers as the feature extractor [79]. Inspired by [41], we use *conv3, conv4 and conv5* layers. Higher layers contain more semantic information while lower layers pay more attention to spatial details. In order to integrate the power of different layers effectively, each layer is convolved with the dual correlation filter to generate a response map. Since the response map of each layer is of different size caused by maxpooling, they are resized to the template size with bilinear interpolation. The final response map is obtained by stacking all the response maps with different weights [41].

In addition, the pre-trained CNN features put more emphasis on invariance representation of semantics with loss of spatial information, during the experiments, it is found that establishing the scale pyramid using CNN feature as in Sec. 4.3.3 cannot estimate target scale accurately but with high computation load. Therefore, we train a separate CF only using HOG feature to estimate the target scale, while using CF of CNN feature to estimate target translation. We find this method works well in practice.

## 5.4 Implementations and Method Analysis

### 5.4.1 Tracking by Multi-Expert Framework

Under the multi-expert framework modeled by discrete graph, object tracking is conducted as follows. The snapshots in the expert ensemble $\mathbf{E}$ are stored chronologically at intervals of $\Delta t$ frames. For efficiency purpose, maximum $N_{\mathbf{E}}$ experts (including the current tracker) are maintained. If the number of experts exceeds $N_{\mathbf{E}}$, the oldest expert is discarded. The update procedure of the multi-expert framework is shown in Fig. 5.2. At each frame, the expert ensemble proposes the potential target candidates $X$. After that, unary scores of the proposed framework are calculated by Eq. 4.3, and binary compatible scores are computed according to Sec. 5.3.2. The best expert hypothesis is then selected by figuring out the path with the highest score defined by Eq. 5.2, and the target state is decided by the best expert. Note that only expert $\mathcal{T}_T$ is updated, so the whole algorithm has low computation.

For HSME-SVM, only target translation is considered. For HSME-CF and HSME-deep, target scale is estimated according to the method described in Section 4.3.3 and 5.3.3 respectively. Generally, scale variation is much smaller than that of translation. For computation efficiency, the scale estimation is only conducted on expert $\mathcal{T}_T$ (current tracker). We find this strategy very effective in practice. Note that in HSME-deep, the CNN feature extraction takes the major part of computation load. However, this forward pass process is operated only once at each frame, since the CNN feature extractor and CF classifier is separated, and only CF is updated at each time step.

The whole HSME tracking algorithm flowchart is summarized in the table below.

**Algorithm 4:** HSME Tracker

```
input  : Initial target bounding box $\mathbf{b}_1$
output : Estimated target state $\mathbf{b}_t = (\hat{x}_t, \hat{y}_t, \hat{s}_t)$
$\mathbf{E} \leftarrow \mathcal{T}_1$;
Initialize the multi-expert graph as in Fig. 5.2;
for $t$ = 1 : $N_{frames}$ do
    Get the target candidate set $X$ proposed by $\mathbf{E}$;
    for $\mathcal{T} \in \boldsymbol{E}$ do
        if $\mathcal{T} = \mathcal{T}_t$ then
            Build the target pyramid at $(\hat{x}_{t-1}, \hat{y}_{t-1})$;
            Compute the response pyramid, estimate the target location $(x_{\mathcal{T}_t}.y_{\mathcal{T}_t})$ and
            scale $\hat{s}_t$;
        else
            Get the response map and estimate the target location $(x_{\mathcal{T}}, y_{\mathcal{T}})$;
        Evaluate the graph unary and binary score according to Sec. 5.3.2;
    Select $\mathcal{T}^* \in \mathbf{E}$ according to Sec. 5.3.1;
    Output the target state $\mathbf{b}_t = (x_{\mathcal{T}_t^*}, y_{\mathcal{T}_t^*}, \hat{s}_t)$;
    if mod $(t, \Delta t) == 0$ then
        $\mathbf{E} \leftarrow \mathbf{E} \cup \mathcal{T}_t$;
        discard the oldest expert when $|\mathbf{E}| > N_{\mathbf{E}}$;
    Re-train the current expert $\mathcal{T}_t$;
    Update the multi-expert graph as in Fig. 5.2;
```

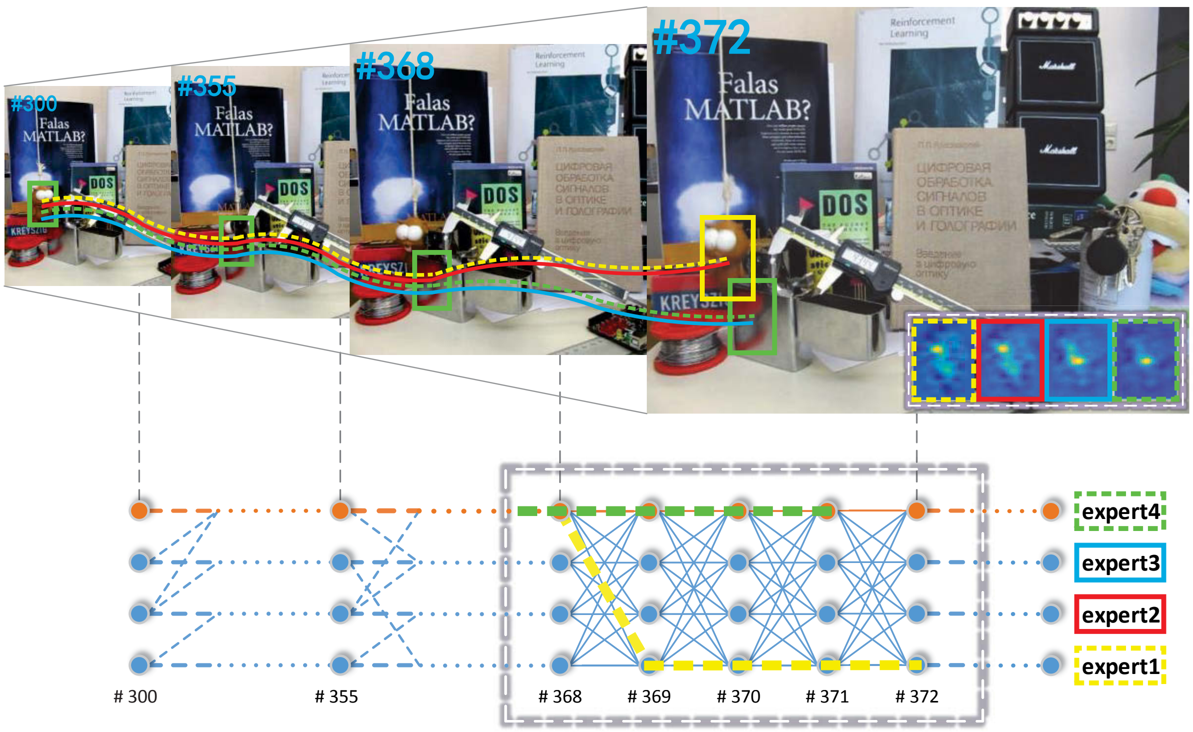


Fig. 5.3 Illustrations for Drift Correction by the proposed graph based multi-expert framework. Four experts are used, and they are stored in chronological order whose trajectories are colored by yellow, red, blue and green respectively (The green one denotes the current tracker). The current tracker tends to drift, and the multi-expert framework corrects the drift by selecting the historical tracker hypothesis colored by yellow. The corresponding expert selection results are shown in the graph. Note the framework finds the disagreement in frame 368 and corrects the drift in frame 372. The response maps of the experts in frame 372 are shown at the bottom-right corner of that image.

### 5.4.2 Multi-Expert Framework Efficiency

In this section, we analyze how the proposed framework correct the drift and its advantages over heuristic multi-expert framework. As a typical example shown in Fig. 5.3, the current tracker tends to drift after the object being occluded, and its trajectory colored by green begins to be attracted by the distractor. However, the drift can be detected by the multi-expert framework. After calculating the path of the highest score in Frame 372, the true object trajectory is recognized, and the drift is corrected by the historical tracker. In the MEEM tracker [55] and SME tracker, the best expert is selected with the highest accumulated score for numbers of the latest frames, where the frame number is a pre-defined parameter, while our proposed framework omits this parameter setting.

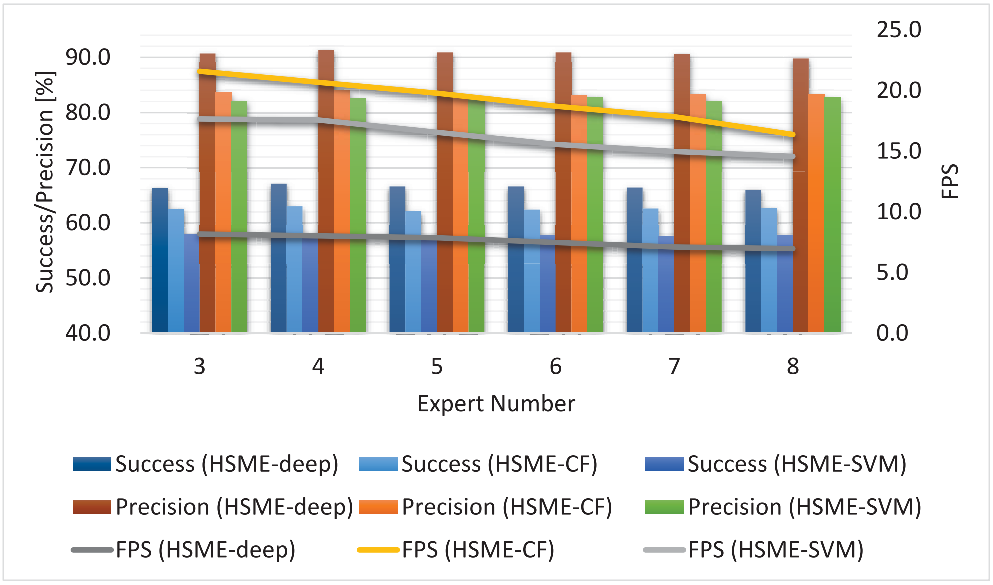


Fig. 5.4 The success/precision scores and FPS of HSME-deep, HSME-CF and HSME-SVM when the expert number varies. The bar plots show the success/precision scores of the three trackers with y-axis on the left, while the curve plots show the FPS of the three trackers with y-axis on the right.

To be more specific, under the proposed framework, the multiple experts usually give very similar hypotheses of the target state when the object is tracked stably. Therefore, in this situation, the multi-expert gives object trajectories approximate to the current tracker. However, when there is disagreement among the expert ensemble, the multi-expert framework is capable of selecting the most reliable object trajectory implicitly by estimating all their recent hypotheses. Moreover, with the dynamic programming solver, the above advantage can be obtained by only evaluating the expert scores of the current frame. Assuming there is $|\mathbf{E}|$ experts at each frame, the computation complexity takes about $\mathcal{O}(|\mathbf{E}|^2)$ time. Note the backward tracing step in Eq. 5.3 is not necessary in real tracking process.

### 5.4.3 Multi-Expert Framework Robustness

In this section, the robustness of the proposed framework is tested by analyzing the three main parameters that may influence the tracking performance.

One of the main parameters of the proposed framework is the number of experts $N_{\mathbf{E}}$. Since the historical tracker snapshots can be thought of the memory storing the target and the background information, intuitively, increasing the number of experts will enlarge the information content. On the other hand, increasing the expert number will result in efficiency

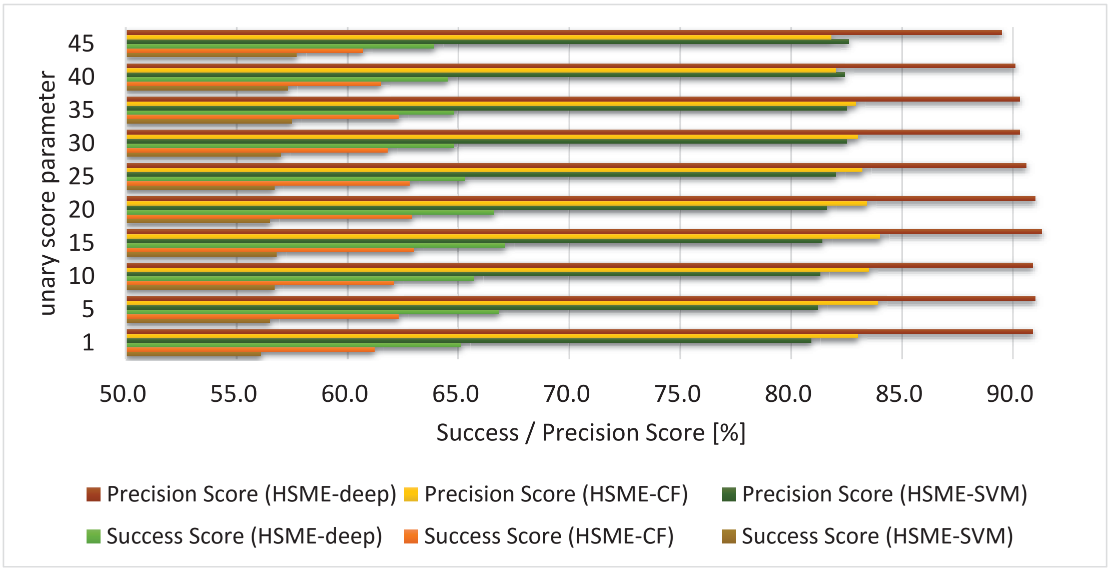


Fig. 5.5 The success/precision scores of HSME-deep, HSME-CF and HSME-SVM when the unary score parameter varies.

decline. Therefore, we conduct the experiments to explore how expert number influences the tracking performance. We test the tracking performance of HSME-deep, HSME-CF and HSME-SVM on OTB-50 dataset using the OPE evaluation [2], and plot their success and precision scores jointly with their frames per second (FPS) variation. We test the expert number from 3 to 8, and the experimental results are shown in Fig. 5.4. From the figure, it is observed that tracking performances are relatively stable with the expert number varies. However, the FPS criterion falls more significantly when the expert number increases. The FPS of HSME-deep declines more gently than HSME-CF and HSME-SVM since most of the time is taken on CNN feature extraction, and the time for forward passes on CNN is the same as the expert number increases. Moreover, the performance scores reach the highest from number of 3 to number of 4, and drop slightly in number of 5. The scores begin to increase from number of 5, and improve gently afterwards. Taken the FPS into consideration. we set the expert number as 4 in this work since both the performance and efficiency are satisfactory at this point.

The second important parameter is the trade-off coefficient $\eta$ of the unary graph score in Eq. 4.3. The parameter $\eta$ indicates what the weight for the likelihood and entropy prior should be to describe the reliability of multi-expert composed by a specified base tracker. Intuitively, the importance of the two terms in Eq. 4.3 should not be the same since different base tracker presents different property. The experiment for unary score parameter $\eta$ is conducted, and Fig. 5.5 shows the performance of the three proposed trackers when $\eta$ increases from 1 to 45. From the test results, it is observed that with $\eta$ increasing, the scores of correlation filter

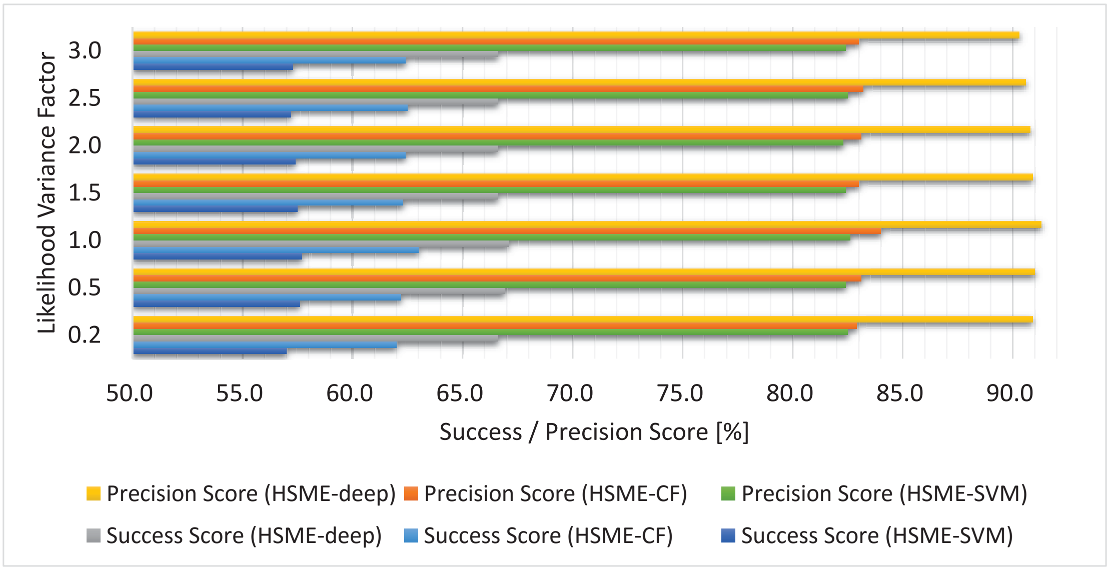


Fig. 5.6 The success/precision scores of the proposed three trackers when the likelihood variance factor varies. Refer to Sec. 5.4.3 for details.

based ensemble trackers (HSME-deep and HSME-CF) first rise, and reach highest when $\eta$ is around 15, then decline slightly when $\eta$ becomes bigger. It also can be seen that when the weight of entropy prior becomes very high ($\eta$ is larger than 35), the performances of HSME-deep and HSME-CF decline dramatically. However, the performance of HSME-SVM rises monotonously when $\eta$ grows. This is because that the energy of response map estimated by correlation filter tracker is focused on the peaks, so that the likelihood term plays a more important role than entropy prior, and a proper trade-off of the two terms presents satisfactory tracking performance. On the other hand, the response map given by the grid search based SVM tracker is more ambiguous than that of correlation filter, which favors the entropy prior to describe the reliability of the base tracker. As a result, we choose $\eta = 15$ for HSME-deep and HSME-CF, and $\eta = 45$ for HSME-SVM.

The third parameter is the variance $\sigma$ of the affinity matrix in Eq. 4.4, which constitutes the likelihood term in the graph score. In HSME-deep and HSME-CF, this parameter describes the similarity of the expert hypotheses. In order to test whether $\sigma$ shows different influence with different target size, we define the likelihood variance factor $f$ that is the proportional to the target size $s$, which we set it as the multiples of $3\sigma$ for convenience according to the 3-sigma rule of Gaussian distribution, i.e. $s = 3\sigma \times f$. Fig. 5.6 shows the success and precision scores varies as $f$ varies from 0.2 to 3. Note the highest score is $f = 1$. Then the scores decline with $f$ increases. Therefore, we set $f = 1$.

## 5.5 Experiments

In this section, we evaluate the proposed three trackers HSME-deep, HSME-CF and HSME-SVM on three large datasets, the first is the 50 sequences Object Tracking Benchmark (OTB-50) [2], the second dataset OTB-100 [124] extends OTB-50 to 100 sequences, the third dataset is the 60 sequences VOT2015 Challenge dataset [121, 11]. Moreover, we also compare the proposed trackers to their base trackers to validate the effectiveness of the proposed framework.

On OTB-50 dataset, we first compare the proposed trackers with state-of-the-art trackers to demonstrate their excellent performance. Then the experiments for their robustness to initialization are conducted according to two criteria: temporal robustness evaluation (TRE) and spatial robustness evaluation (SRE). Lastly, the trackers are tested on sequences of eight main attributes to analyze their performances in different scenarios. Moreover, we also compare the proposed trackers to their base trackers to validate the effectiveness of the proposed framework. In addition, the proposed trackers are evaluated on the recent OTB-100 dataset to test their performances on relatively large dataset. To further verify the efficiency of our trackers, we test the three trackers on VOT2015 dataset, and the experimental results are reported for evaluation.

### 5.5.1 Experimental Setup

#### Experimental Platform and Metrics

The proposed trackers, HSME-deep, HSME-CF and HSME-SVM are implemented in Matlab&C, where HSME-deep is based on MatConvNet toolbox [134]. HSME-CF and HSME-SVM run at roughly 20 fps and 15 fps respectively on the 3.20GHz CPU with 8GB RAM. HSME-deep runs about 8 fps with NVIDIA Tesla K20c 5GB GPU. Although with multiple experts, the proposed trackers are with high efficiency, mostly due to the efficiency of the dynamic programming solver and the low computation load of the base trackers, especially the correlation filter.

The experimental metrics for OTB-50 have been introduced in Chapter 3, and metrics for OTB-100 are the same as OTB-50. Metrics for VOT2015 has been elaborated in Chapter 4.

#### Parameter Setup

The parameters setup of the multi-expert framework is as follows. The max number of experts $N_{\mathbf{E}}$ is set to 4. The frame interval $\Delta t$ is set to 50, and we find the performance is not sensitive to tens of frames such as 30 or 40. The trade-off parameter of the graph score in Eq. 5.1 is set to $\beta = 1$ for simplicity. The trade-off parameter $\eta$ in Eq. 4.3 is set to 15 for HSME-deep /

HSME-CF, and 45 for HSME-SVM. The parameters for target candidate set $X$ are the same as in Sec. 4.4.

For the online SVM base tracker, we set parameters the same as those in [55] for fair comparisons in the experiments. For the correlation filter base tracker and the scale target pyramid, parameters are set the same as in Sec. 4.4. For HSME-deep, the weights for stacking response maps are set to 1, 0.5 and 0,02 for *conv5, conv4* and *conv3* respectively similar to [41].

### 5.5.2 Evaluation on OTB-50 Dataset

Besides the trackers provided by the benchmark [2], we also compare our methods with many state-of-the-art trackers, including STCT [47], HCT [41], Staple [126], TGPR [135], and the trackers MEEM [55], CN [33] and KCF [9] that have been compared with in the last chapter. Among the compared trackers, STCT and HCT are based on Convolutional Neural Networks. Here, we also compare the TVML and SME methods proposed in this thesis.

#### Overall Performance

According to the evaluation methods by OTB-50, the one-pass evaluation (OPE) performance is illustrated in the Success Plot and Precision Plot shown in Fig. 5.7.

As shown in the plot, HSME-deep achieves 67.1% success rate and 91.3% precision rate, both of which rank first among all the compared trackers. HSME-CF gets 63% success rate and 84% precision rate, which ranks only behind the CNN trackers, STCT and HCT.

It can be seen that the proposed three trackers in this chapter performs better than TVML and SME. While HSME-CF and SME adopt the same base tracker, the former method exceeds the latter by 2.3% and 2.0% in success and precision scores respectively, which validate the discrete graph based tracking framework works better than the heuristic method. Particularly, MEEM is also based on historical tracker ensemble, and it has the same online SVM base tracker as the proposed HSME-SVM. Compared to MEEM, HSME-SVM improves the performance results significantly by 2.5% of the AUC score and 2.9% of the CLE score, which shows the proposed framework works better than the original framework based on simple accumulated score.

The proposed HSME-deep tracker further boosts the performance, and it surpasses HSME-CF and HSME-SVM with large margin, especially exceeds in the CLE score by 8.6% compared to HSME-CF. HCT is also the correlation filter based tracker with CNN feature, which can be regarded as our base tracker. Compared to HCT, HSME-deep improves the success score by 10.9%. The overall plot demonstrates the proposed trackers are effective and promising.

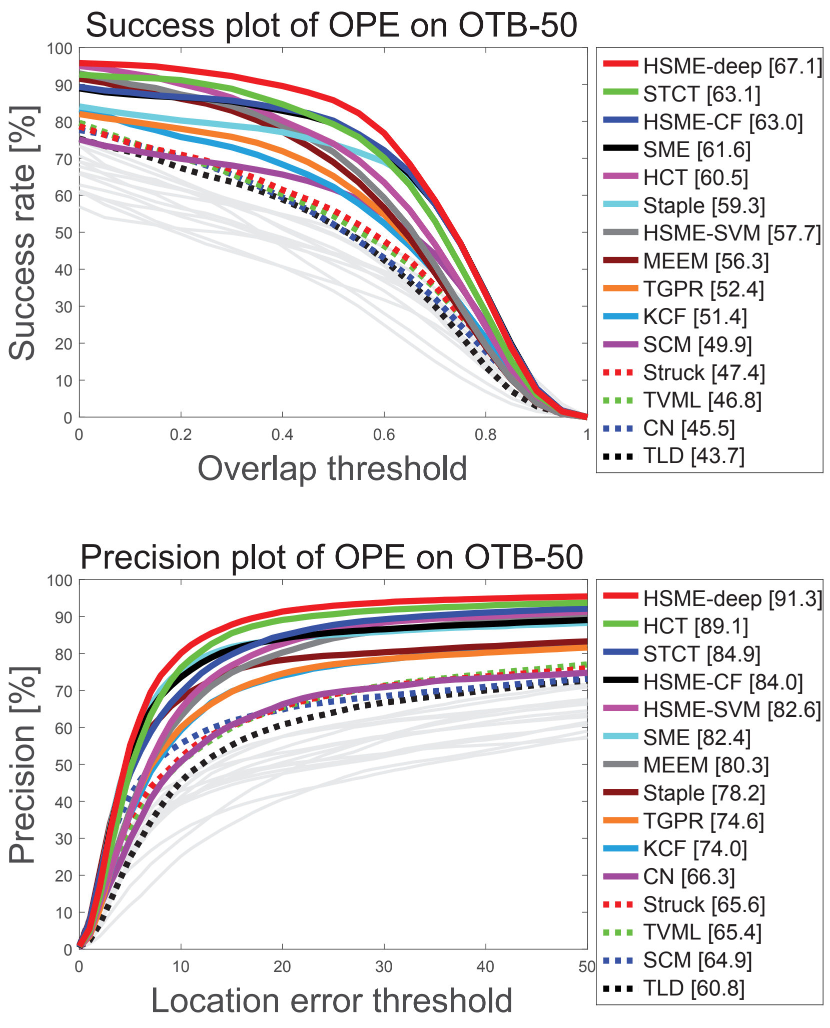


Fig. 5.7 The success plots and precision plots over OTB-50 dataset using one pass evaluation (OPE). The legend illustrates the area under curve (AUC) for the success plot, and the score of the threshold 20 for the precision plot. Only scores of the top-15 trackers are shown, and the others are plotted in light gray curves.

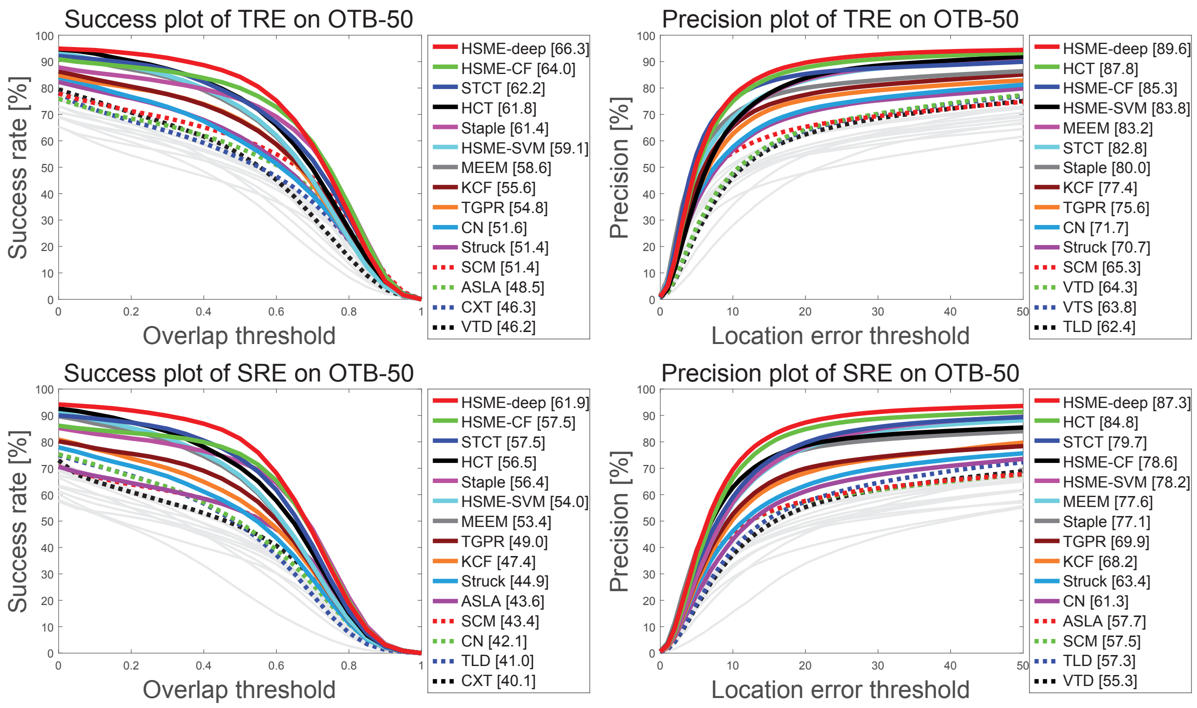


Fig. 5.8 The success plots and precision plots over OTB-50 dataset using temporal robustness evaluation (TRE) and spatial robustness evaluation (SRE).

### Robustness to Initialization

Object trackers are usually sensitive to different initializations. In this part, we evaluate the robustness to initialization of the proposed methods according to the strategy in [2]. Two evaluation strategies are used: temporal robustness (TRE) and spatial robustness (SRE).

In TRE criterion, each sequence is divided into 20 segmentations, and the test tracker runs from the beginning of each segmentation with its ground truth initialization to the end of the image sequence. The SRE scheme tests a tracker from different positions of initialization close to the ground-truth in the first frame, and the procedure repeats 12 times for each sequence.

The TRE and SRE test results for OTB-50 are shown in Fig. 5.7. From the plot, the proposed HSME-deep ranks first in the two tests for both the success plot and precision plot, and HSME-CF ranks second in both the success rates. It can also be found that HSME-CF is robust to different initialization, which performs even better than the CNN trackers in both success plots.

### Attribute Performance

In this experiment, the benchmark sequences are divided into 8 main attributes to evaluate the trackers in different scenarios. We choose the top-10 trackers (besides SME) in the OPE test

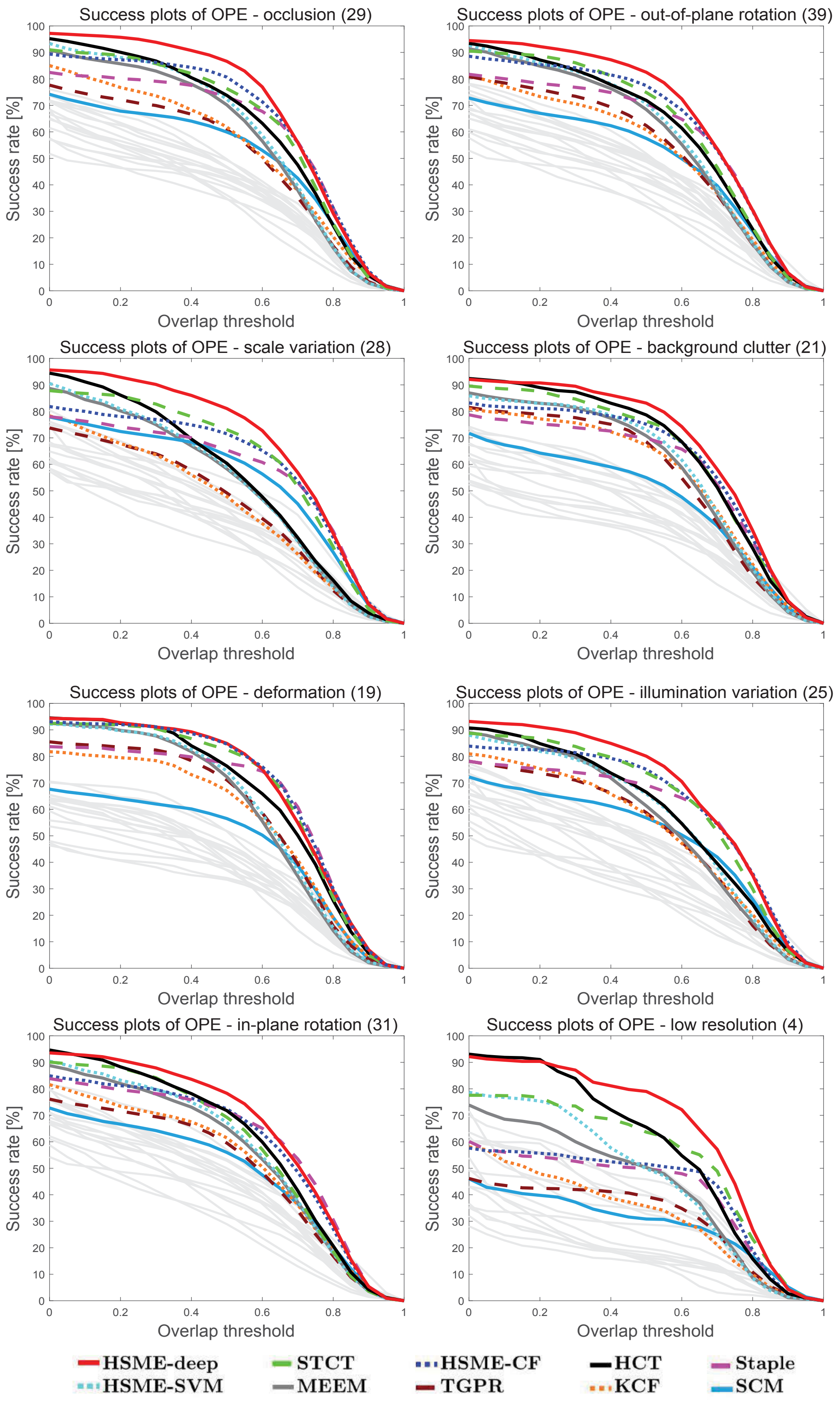


Fig. 5.9 Success plots for eight attributes of the top-10 performance trackers on OTB-50.

Table 5.1 Scores of the attribute plots in Fig. 5.9.

| | HSME-deep | STCT | HSME-CF | HCT | Staple | HSME-SVM | MEEM | TGPR | KCF | SCM |
|---|---|---|---|---|---|---|---|---|---|---|
| **OCC** | 67.2 | 61.2 | 62.8 | 60.6 | 58.5 | 57.5 | 55.8 | 48.3 | 51.4 | 48.7 |
| **OPR** | 64.4 | 59.8 | 61.0 | 58.7 | 56.9 | 56.6 | 55.4 | 50.4 | 49.5 | 40.0 |
| **SV** | 65.5 | 59.4 | 57.4 | 53.1 | 54.5 | 50.6 | 50.2 | 42.6 | 42.7 | 51.8 |
| **BC** | 65.3 | 61.0 | 59.3 | 62.3 | 55.7 | 56.6 | 55.7 | 52.9 | 53.5 | 45.0 |
| **DEF** | 65.6 | 64.4 | 65.8 | 62.6 | 60.7 | 57.9 | 57.1 | 55.4 | 53.4 | 44.8 |
| **IV** | 64.3 | 60.2 | 60.0 | 56.0 | 56.1 | 53.9 | 52.6 | 48.2 | 49.3 | 47.3 |
| **IPR** | 62.5 | 56.5 | 57.2 | 58.2 | 57.6 | 54.7 | 53.6 | 47.6 | 49.7 | 45.8 |
| **LR** | 62.5 | 52.5 | 40.8 | 55.7 | 39.5 | 44.4 | 41.6 | 30.1 | 31.2 | 27.9 |

red: rank 1, green: rank 2, blue: rank 3

to further evaluate their performance in the attribute test. As most of the proposed trackers are target scale adaptive, the AUC score measures the tracker performance more accurately than CLE that is with one threshold, so the success plot is the main analysis evaluation tool. Therefore, we report the eight main attributes of success plots in Fig. 5.9, whose scores are summarized in Table 5.1.

As illustrated in the plots, the proposed trackers rank first in all the attributes. Particularly, in seven of eight attributes, HSME-deep ranks first, and exceeds the second one with large margin. Among all the attributes, the scale variation performance is improved significantly, which shows our scale scheme is very effective. In addition, HSME-CF also gets more favorable scores than other correlation filter based trackers, Staple [126] and KCF [9], which demonstrates the effectiveness of the multi-expert framework. HSME-SVM performs favorably against the counterpart MEEM, especially in low resolution sequences with significant improvement.

### Framework Effectiveness Validation

In this experiment, we compare each of our proposed trackers with their base trackers to evaluate the effect of the proposed multi-expert framework. We summarize their success and precision scores in Table 5.2. From the table, both the success and precision scores of the three proposed trackers have significant improvement compared with their base trackers. From the success score perspective, the improvement on CF is higher than that of SVM, which is majorly due to the likelihood term that is more suitable for CF base trackers. Besides, the results of HSME-SVM and HSME-SVM-base demonstrate that the proposed framework pays more attention to correcting the translation estimation error by selecting the reliable expert.

Table 5.2 Success and precision scores for the proposed trackers and their base trackers. (Darker cells denote higher scores.)

| | HSME-deep | HSME-deep-base | HSME-CF | HSME-CF-base | HSME-SVM | HSME-SVM-base | |
|---|---|---|---|---|---|---|---|
| OPE | 67.1 | 63.3 | 63.0 | 57.4 | 57.7 | 55.8 | Success |
| TRE | 66.3 | 64.6 | 64.0 | 58.5 | 59.1 | 57.4 | |
| SRE | 61.9 | 59.8 | 57.5 | 56.5 | 54.0 | 52.2 | |
| OPE | 91.3 | 87.1 | 84.0 | 80.4 | 82.6 | 80.3 | Precision |
| TRE | 89.6 | 87.8 | 85.3 | 82.6 | 83.8 | 81.2 | |
| SRE | 87.3 | 84.7 | 78.6 | 77.1 | 78.2 | 75.6 | |

Table 5.3 AUC \CLE scores for long term sequences shown in Fig. 5.11.

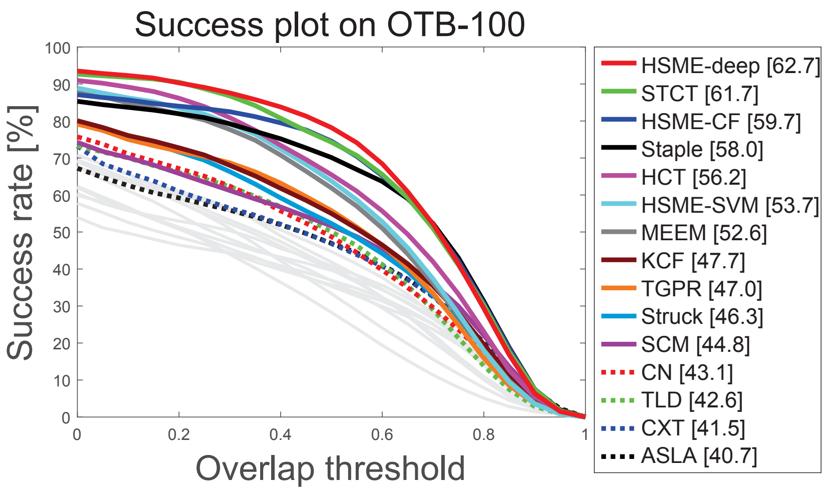
| *Sequence* | *Frames* | **HSME-deep** | **HSME-CF** | **HSME-SVM** | **STCT** | **HCT** | **Staple** | **MEEM** |
|---|---|---|---|---|---|---|---|---|
| *Car24* | 3059 | 0.73 \ 4.32 | 0.71 \ 5.13 | 0.43 \ 7.52 | 0.68 \ 5.39 | 0.41 \ 7.93 | 0.43 \ 5.85 | 0.43 \ 7.48 |
| *RedTeam* | 1918 | 0.47 \ 3.69 | 0.46 \ 6.36 | 0.51 \ 4.06 | 0.52 \ 4.71 | 0.46 \ 4.80 | 0.57 \ 3.04 | 0.50 \ 4.31 |
| *Girl2* | 1500 | 0.61 \ 31.73 | 0.55 \ 33.72 | 0.60 \ 19.30 | 0.07 \ 294.07 | 0.07 \ 118.63 | 0.11 \ 114.12 | 0.57 \ 36.10 |
| *Sylvester* | 1345 | 0.77 \ 4.31 | 0.69 \ 9.32 | 0.67 \ 9.33 | 0.65 \ 9.72 | 0.65 \ 12.68 | 0.56 \ 14.16 | 0.66 \ 10.01 |
| *Box* | 1161 | 0.76 \ 8.50 | 0.75 \ 9.73 | 0.66 \ 12.70 | 0.72 \ 11.33 | 0.28 \ 107.23 | 0.35 \ 90.85 | 0.63 \ 15.60 |
| *Mean* | \ | 0.67 \ 10.51 | 0.63 \ 12.85 | 0.57 \ 10.58 | 0.53 \ 65.04 | 0.37 \ 50.25 | 0.40 \ 45.60 | 0.56 \ 14.7 |

red: rank1, green: rank 2, blue: rank 3; AUC \ CLE

## 5.5.3 Evaluation on OTB-100 Dataset

### Overall Performance

We further compare our methods with the trackers mentioned in Sec. 5.5.2 on the recently introduced OTB-100 dataset to validate their performances on relatively large dataset. As illustrated in Fig. 5.10, the top-15 trackers are colored in the plots, and the remainders are only drawn in light gray color. The evaluation criteria of the success plot and the precision plot are also the previously mentioned AUC and CLE score. As shown in the plots, the proposed HSME-deep ranks first in both the success and precision criteria. And the scores of HSME-CF are only lower than CNN based trackers, STCT and HCT. HSME-SVM also ranks higher than MEEM in both plots.



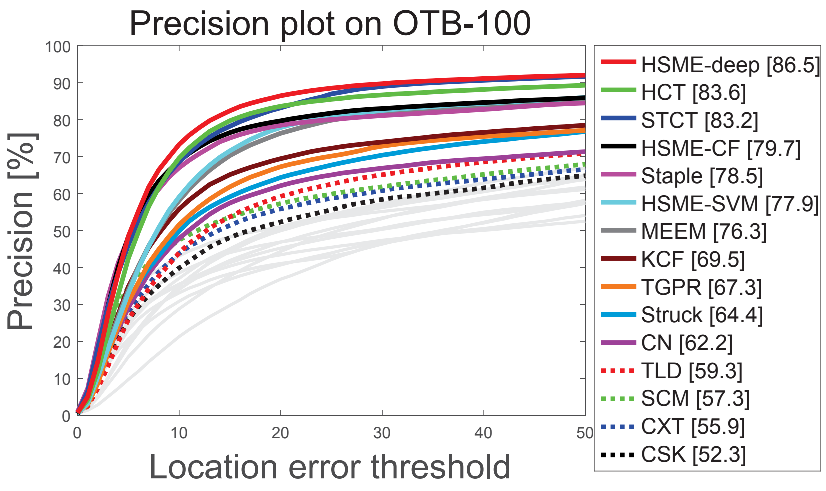


Fig. 5.10 The success plot and precision plot over OTB-100 dataset using one pass evaluation (OPE). Legends for tracker scores are located beside each plot. Only top-15 trackers are colored, and the others are shown in gray.

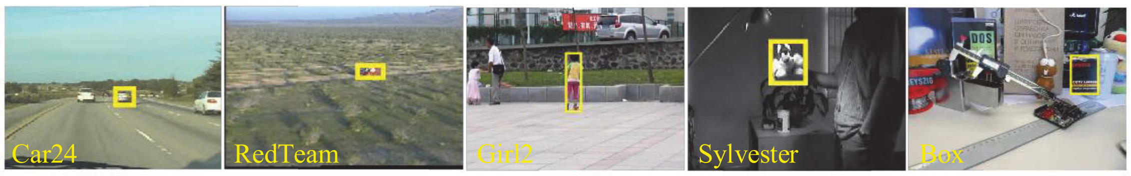


Fig. 5.11 Long term sequences snapshots. Sequence name is at the lower left corner.

### Long Term Sequences

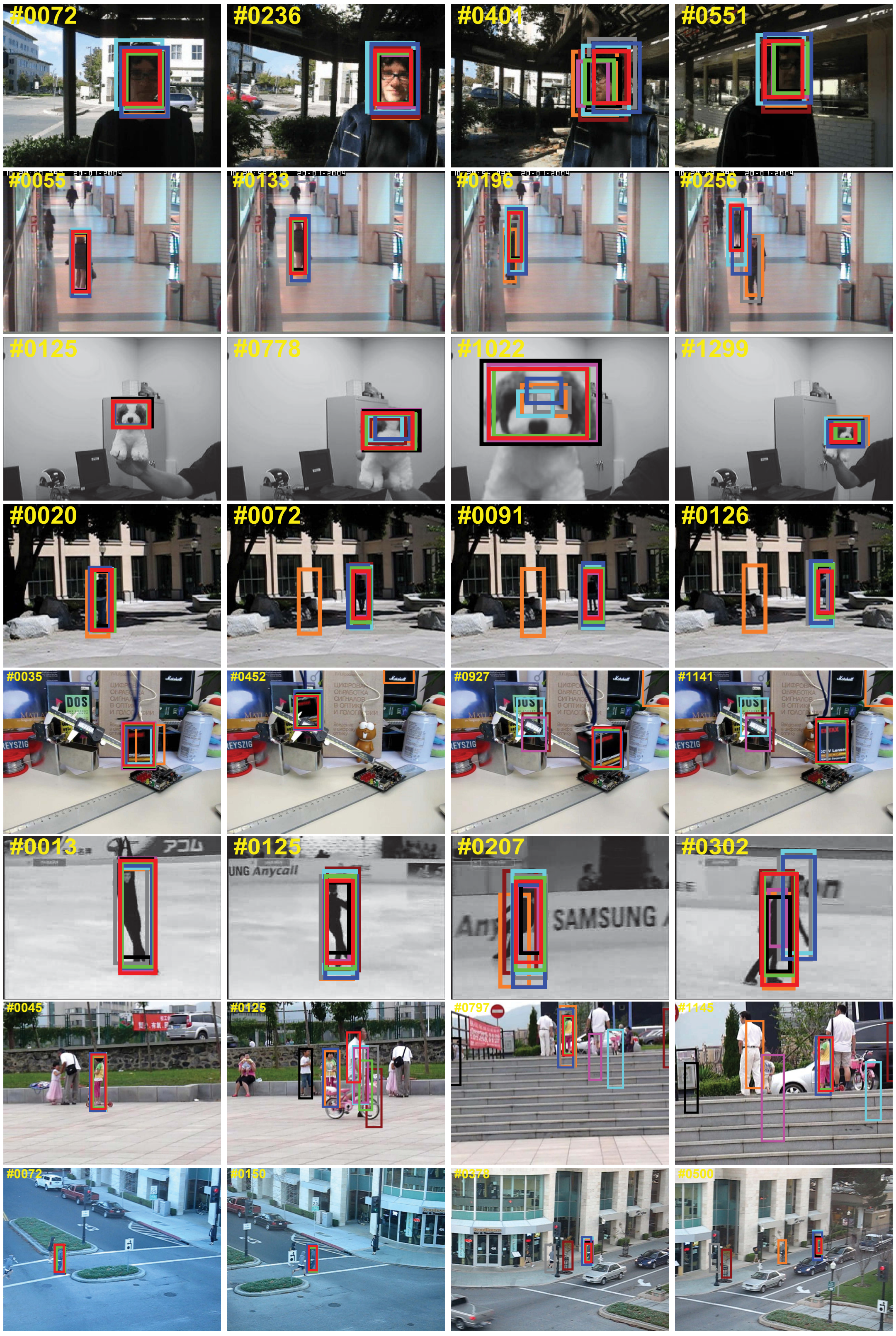
In this section, we test the proposed trackers with the top trackers in the overall performance on the relatively long sequences (at least $> 1000$ frames). The test sequences are shown in Fig. 5.11. *Car24* and *Girl2* are the videos of real scenarios, and *RedTeam* is the aerial video, while *Sylvester* and *Box* are normal tracking test sequences. The metrics are the average AUC (in decimal) and CLE in this test. The results are shown in Table 5.3. From the table, HSME-deep ranks first for both the AUC and CLE mean measurements. It is also observed that for long term sequences, the multi-expert framework is capable of improving the tracking performance, especially in *Girl2*, with frequent occlusions, and deformations, even the CNN trackers, STCT and HCT, fail to track the target, while our trackers with the HSME framework can deal with this situation.

### Typical Results Analysis

Some typical tracking results of the top trackers are shown in Fig. 5.12. Among all the test sequences, *Dog1* and *Box* have significant scale variations. *Walking2*, *Girl2*, *Human4*, *Jogging* and *Skating1* go through part or whole occlusion. In addition, *Skating1* has illumination changes due to the stage light. Some targets in the sequences suffer from frequent appearance variations and non-rigid appearance changes, such as in *Box*, the object has multiple views, and in *Skater2*, the skater has large non-rigid appearance transformations, the same challenge can be found in *Human8*. *Skiing* and *MotorRolling* have severe deformations, which are very hard to track. From *Dog1* and *Box*, we can see that HSME-deep and HSME-CF perform well in handling scale variation. Especially in *Dog1* with large scale changes in frame 1022 and frame 1299, the proposed trackers estimate the scale variations accurately.

At the last stage of *Skating1*, where most of the compared trackers fail, HSME-CF is able to catch the target despite of its significant illumination changes. This is because our tracker combines both the HOG and color feature, so as to adapt to the object appearance variations caused by outer factors. When there is large object appearance changes, like in frame 401 of *Trellis*, most of the compared algorithms start to drift, but our tracker is capable of dealing with the challenge. In *Skiing*, where most of the compared trackers lose the target, HSME-deep, HSME-SVM and HCT are capable of catching the object. It is also noted that in *MotorRolling*, only CNN feature based trackers, HSME-deep, STCT and HCT can track the target because of the high rotation deformation.

There are also the results to demonstrate the superiority of HSME-SVM over MEEM. In *Girl2*, when an adult walks in front of the girl, MEEM and many other trackers are drifted by the occlusion, while HSME-SVM corrects the drift and continues to track the truth target. The


#0072
#0236
#0401
#0551
#0055
#0133
#0196
#0256
#0125
#0778
#1022
#1299
#0020
#0072
#0091
#0126
#0035
#0452
#0927
#1141
#0013
#0125
#0207
#0302
#0045
#0125
#0797
#1146
#0072
#0150
#0378
#0500

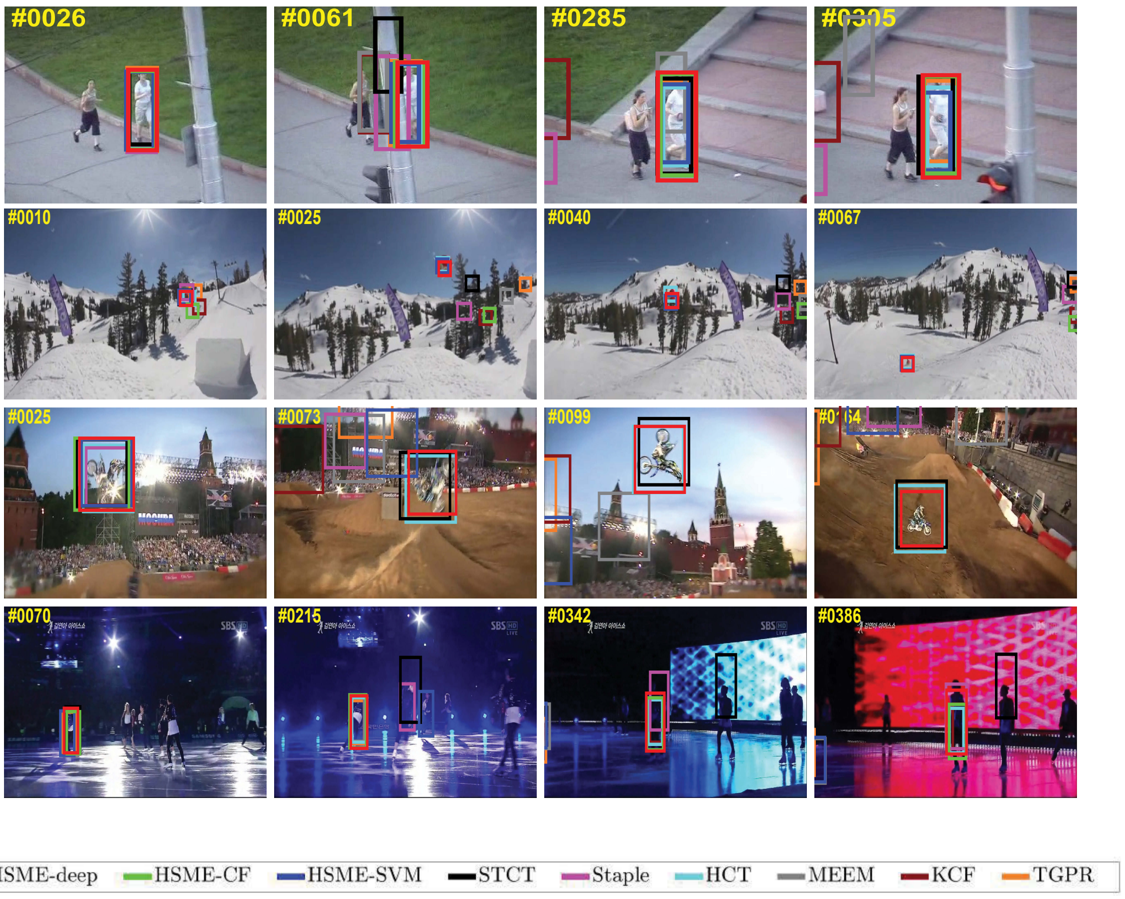


Fig. 5.12 Tracking results of the top nine algorithms (HSME-deep, HSME-CF, HSME-SVM, STCT [47], Staple [126], HCT [41], MEEM [55], KCF [9], TGPR [135]) on TB-100 over twelve typical sequences. The video illustrations from top to bottom are *Trellis, Walking2, Dog1, Human8, Box, Skater2, Girl2, Human4, Jogging, Skiing, MotorRolling and Skating1*.

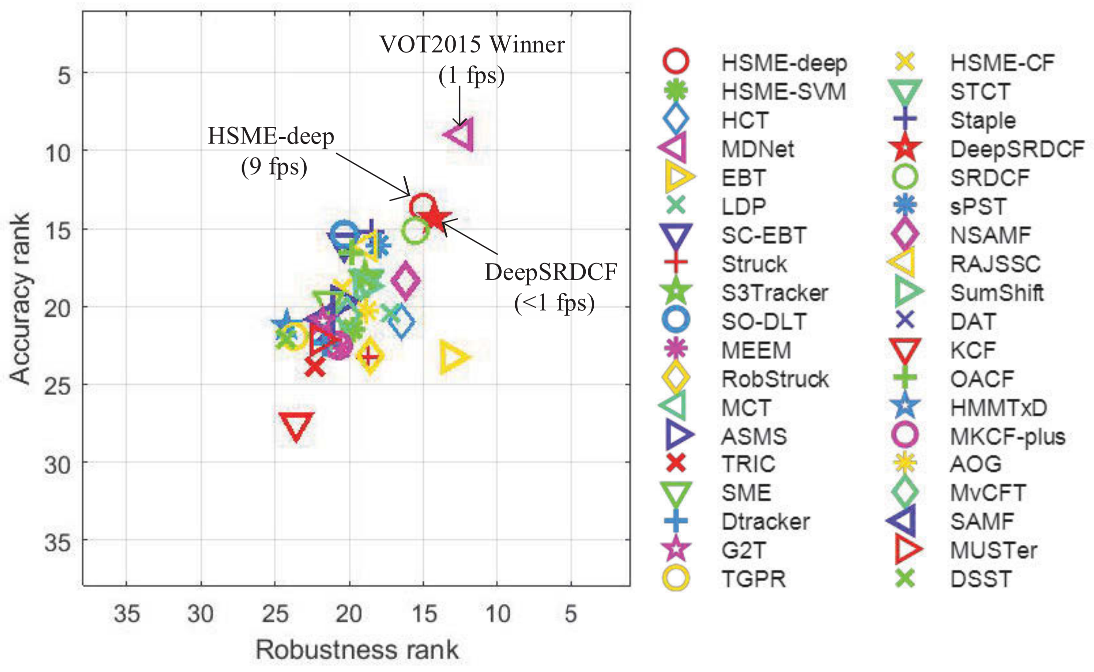


Fig. 5.13 The AR ranking plot for VOT2015 Dataset. The tracker is better if its legend resides closer to the top-right corner of the plot.

same phenomenon can be observed in the *Walking2* sequence. In *Human4*, MEEM is drifted by the distractor, while HSME-SVM continues to track the target, which also demonstrates the advantage of our framework to that of MEEM. The same conclusion can also be drawn from sequence *Jogging*.

However, we can also see the scenario that all the trackers cannot well deal with, such as in *Skater2*, although all the trackers can catch the target, no one is able to locate the object accurately due to the frequent non-rigid transformations of the skater. We hope this challenge can be solved by further improving the multi-expert framework in the future.

### 5.5.4 Evaluation on VOT2015 Challenge Dataset

The number of sequences in VOT2015 Challenge Dataset has been enlarged to 60 compared to VOT2013 and VOT2014, whose numbers of sequences are 16 and 25 respectively [11]. Besides the trackers participated in the VOT2015 challenge, we also compare with 3 other state-of-the-art trackers, including STCT [47], HCT [41], Staple [126].

Table 5.4 The results of VOT2015 Challenge Dataset.

| | **Accuracy** | **Failures** | **Overall Rank** |
|---|---|---|---|
| MDNet | 0.56 | 0.72 | 10.67 |
| DeepSRDCF | 0.53 | 1.12 | 14.30 |
| **HSME-deep** | 0.53 | 1.24 | 14.33 |
| SRDCF | 0.52 | 1.31 | 15.34 |
| sPST | 0.51 | 1.29 | 16.82 |
| Staple | 0.52 | 1.56 | 17.04 |
| NSAMF | 0.49 | 1.31 | 17.29 |
| RAJSSC | 0.52 | 1.66 | 17.42 |
| SO-DLT | 0.54 | 1.81 | 17.85 |
| SC-EBT | 0.52 | 1.37 | 18.14 |
| OACF | 0.52 | 1.84 | 18.18 |
| EBT | 0.45 | 1.05 | 18.21 |
| HCT | 0.47 | 1.27 | 18.51 |
| STCT | 0.51 | 1.56 | 18.59 |
| S3Tracker | 0.47 | 1.80 | 18.71 |
| LDP | 0.45 | 1.36 | 18.83 |
| SumShift | 0.47 | 1.75 | 18.84 |
| AOG | 0.47 | 1.88 | 19.57 |
| **HSME-CF** | 0.47 | 2.04 | 19.65 |
| ASMS | 0.46 | 1.98 | 20.13 |
| MvCFT | 0.48 | 2.24 | 20.36 |
| sme | 0.48 | 2.26 | 20.50 |
| **HSME-SVM** | 0.46 | 2.24 | 20.62 |
| RobStruck | 0.44 | 1.97 | 20.86 |
| SAMF | 0.43 | 2.24 | 21.63 |
| DSST | 0.49 | 2.31 | 20.99 |
| Struck | 0.44 | 2.11 | 21.24 |
| MEEM | 0.44 | 2.41 | 21.37 |
| G2T | 0.43 | 2.25 | 21.64 |
| MKCF-plus | 0.43 | 2.33 | 21.72 |
| DAT | 0.44 | 2.36 | 21.83 |
| MCT | 0.42 | 2.36 | 22.00 |
| Dtracker | 0.43 | 2.38 | 22.04 |
| MUSTer | 0.44 | 2.47 | 22.72 |
| HMMxD | 0.44 | 2.48 | 22.82 |
| TGPR | 0.45 | 2.31 | 23.09 |
| TRIC | 0.44 | 2.34 | 23.18 |
| KCF | 0.43 | 2.53 | 25.54 |

[1]red: rank 1, green: rank 2, blue: rank 3

The accuracy and failures, as well as the overall ranking results for all the tested trackers are listed in Table 5.4. As there is a large number of compared trackers, only the trackers rank higher than KCF [9] are listed.

According to the VOT evaluation criteria [121], the overall experimental results are illustrated in accuracy-robustness (AR) ranking plot, as shown in Fig. 5.13. The AR ranking plot shows average ranking scores of all the sequences for each tracker in the joint accuracy-robustness rank space. The details for the evaluation method can be referred to [11, 121]. From the table and the plot, it is observed that the proposed HSME-deep ranks higher than most of the compared trackers, and only MDNet (the VOT2015 winner) and DeepSRDCF rank higher than HSME-deep. However, the AR ranking plot shows that the speed of the proposed HSME-deep is about 9 fps, which is much faster than MDNet and DeepSRDCF, because MDNet relies on fine-tuning the CNN model and DeepSRDCF needs to solve the optimization problem iteratively online. Besides, HSME-CF ranks well in accuracy, even better than CNN tracker HCT. Finally, HSME-SVM ranks higher than MEEM as well.

## 5.6 Conclusion

In this chapter, we propose an effective discrete graph based multi-expert framework to handle the tracker drift problem. The graph nodes are modeled by the hypotheses of the multiple experts, which are composed of the current tracker and the historical trained tracker snapshots. As the drift tracker tends to be more confident to its own estimation, by extending the work of last chapter, the graph score is defined in a semi-supervised learning manner to describe the relationship of the multiple experts as well as their ambiguity. With the dynamic programming solver, the tracker drift is automatically detected and corrected by analyzing the recent performance of the multi-expert with only evaluating the graph scores of the current frame. Three base trackers, online SVM with a budget, and regression correlation filter (hand-crafted features and CNN features) are integrated into the proposed framework. The experiments are conducted on three large tracking datasets, which demonstrate the proposed trackers perform favorably against state-of-the-art methods.

# Chapter 6

# Conclusions and Future Work

## Conclusions

This thesis focuses on the tracking drift problem. In the first chapter, related work and literature are reviewed in detail, and then the reasons for tracking drift are analyzed carefully. As a result, we handle tracking drift by tackling three issues, including target representation, target feature matching, and tracker online update. In particular, four new methods are proposed as follows:

In order to design robust shape feature to tackle tracking drift, we propose a Weber's Law Shape Descriptor (WLSD), and apply the WLSD into infrared object tracking by designing a multi-feature integration tracking framework including target shape, area and trajectory. The proposed WLSD is tested on MPEG-7 and Tari shape datasets. Although WLSD is the global shape descriptor with low computation load, the experimental results demonstrate the proposed WLSD is as discriminative as many advanced local shape descriptors. We further evaluate the proposed tracking method on infrared videos to validate the multi-feature integration tracking framework is capable of tracking infrared objects successfully.

The second work of this thesis pursues effective target feature matching scheme to handle the tracking drift. In particular, we take advantage of distance metric learning, and propose a new time varying metric learning model for object tracking, which is able to better distinguish the target and background. The experiment is conducted on OTB-50 dataset. The evaluation results validate the proposed method performs better than the existing trackers provided by the benchmark.

Tracker online update usually leads to tracking drift. We propose a tracker snapshot multi-expert tracking method to conquer the tracker online update drift problem, and propose a Scale-adaptive Multi-Expert (SME) tracker accordingly. With the efficient base tracker of correlation filter, the proposed SME tracker is capable of detecting the tracker drift and select the appropriate expert to correct the drift. The experiments on OTB-50 and VOT2015 datasets demonstrate SME is more accurate than most of state-of-the-art trackers while runs at real-time speed.

To further improve the above multi-expert tracker, we propose to use discrete graph to model the multi-expert tracker ensemble. Under the proposed method, the multi-expert selection issue can be regarded as solving the discrete graph optimization problem, which analyzes the reliability of the multi-expert and their estimated trajectories by only evaluating expert scores of the single frame. We integrate three base trackers to the proposed tracking framework to validate its generalization capacity, and evaluate the corresponding three trackers on OTB-50, OTB-100 and VOT2015 datasets extensively. The experimental results demonstrate the effectiveness of the proposed method, as well as the superiority over the compared state-of-the-art trackers.

## Future Work

The future work of this thesis can be extended from many aspects.

In Chapter 2, more advanced target searching framework can be exploited to integrate with the proposed WLSD shape descriptor. In addition, more effective scale selection algorithm can be used to further improve the capacity of WLSD. In Chapter 3, other solvers for the TVML recursive graphical model are expected to estimate the metric more efficiently.

For Chapter 4 and 5, the time interval of multiple experts is set beforehand. The future aim is to set the time interval adaptively online so that the target information contained in the multiple experts can complement each other. From another point of view, the proposed discrete graph based multi-expert selection framework in Chapter 5 can be very general among tracker ensemble methods. A straightforward extension is to design graph score suitable for other type of base trackers besides response map oriented trackers. Furthermore, the discrete graph of single branch tree can be extended to general tree structure, which may be applied to part-based object tracking.

Besides the aforementioned future work, there can also be some potential extensions to combine the proposed methods from different chapters. For instance, the infrared object tracking performance may be further boosted to a large extent when integrating the WLSD descriptor into the TVML model, so as to learn a discriminative metric for WLSD feature.

# References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems (NIPS)*, pages 1097–1105, 2012.

[2] Y. Wu, J. Lim, and M. H. Yang. Online object tracking: A benchmark. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2411–2418, 2013.

[3] A. W. M. Smeulders, D. W. Chu, R. Cucchiara, S. Calderara, A. Denhghan, and M. Shah. Visual tracking: an experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1442–1468, 2014.

[4] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. V. D. Hengel. A survey of appearance models in visual object tracking. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 4(4):58, 2013.

[5] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *Acm computing surveys (CSUR)*, 38(4):13, 2006.

[6] S. Avidan. Support vector tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(8):1064–1072, 2004.

[7] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7):1409–1422, 2012.

[8] B. Babenko, M. H. Yang, and S. Belongie. Robust object tracking with online multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33 (8):1619–1632, 2011.

[9] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2015.

[10] S. Hare, A. Saffari, and P. H. Torr. Struck: Structured output tracking with kernels. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 263–270, 2011.

[11] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, L. Cehovin, et al. The visual object tracking vot2015 challenge results. *Proc. of IEEE International Conference on Computer Vision Workshops*, pages 564–586, 2015.

[12] D. A. Ross, J. Lim, R. S. Lin, and M. H. Yang. Incremental learning for robust visual tracking. *International Journal of Computer Vision*, 77(1-3):125–141, 2008.

[13] J. Kwon and K. M. Lee. Visual tracking decomposition. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1269–1276, 2010.

[14] X. Jia, H. Lu, and M. H Yang. Visual tracking via adaptive structural local sparse appearance model. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1822–1829, 2012.

[15] W. Wei, H. Lu, and M. H. Yang. Robust object tracking via sparsity-based collaborative model. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1838–1845, 2012.

[16] S. J. Julier and J. K. Uhlmann. A new extension of the kalman filter to nonlinear systems. *Symposium of International Aerospace/Defence Sensing, Simulation and Controls*, 3 (26):182–193, 1997.

[17] A. Doucet and A. M Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of Nonlinear Filtering*, 12:656–704, 2009.

[18] A. Doucet, N. D. Freitas, and N. Gordon. An introduction to sequential monte carlo methods. *In Sequential Monte Carlo methods in practice*, pages 3–14, 2001.

[19] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[20] P. Perez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. *Proc. of the European Conference on Computer Vision (ECCV)*, 2002.

[21] T. Liu, G. Wang, and Q. Yang. Real-time part-based visual tracking via adaptive correlation filters. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[22] Y. Li, J. Zhu, and S. C. Hoi. Reliable patch trackers: Robust visual tracking by exploiting reliable patches. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[23] J. Kwon and K. M. Lee. Tracking by sampling and integrating multiple trackers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1428–1441, 2014.

[24] J. Kwon and K. M. Lee. Highly non-rigid object tracking via patch-based dynamic appearance modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(10):2427–2441, 2013.

[25] C. M. Bishop. *Pattern recognition and machine learning*. 2006.

[26] S. Avidan. Ensemble tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(2):261–271, 2007.

[27] K. Zhang, L. Zhang, and M. H. Yang. Real-time compressive tracking. *Proc. of the European Conference on Computer Vision (ECCV)*, pages 864–877, 2012.

[28] H. Grabner, M. Grabner, and H. Bischof. Real-time tracking via on-line boosting. *Proc. of British Machine Vision Conference (BMVC)*, 1:47–56, 2006.

[29] H. Grabner, C. Leistner, and H. Bischof. Semi-supervised on-line boosting for robust tracking. *Proc. of the European Conference on Computer Vision (ECCV)*, pages 234–247, 2008.

[30] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[31] D. S Bolme, J. R Beveridge, B. Draper, and Y. M. Lui. Visual object tracking using adaptive correlation filters. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2544–2550, 2010.

[32] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels. *Proc. of the European Conference on Computer Vision (ECCV)*, pages 702–715, 2012.

[33] M. Danelljan, F. S. Khan, M. Felsberg, and van de Weijer. Adaptive color attributes for real-time visual tracking. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1090–1097, 2014.

[34] M. Danelljan, G. Hager, F. Khan, and M. Felsberg. Accurate scale estimation for robust visual tracking. *Proc. of British Machine Vision Conference (BMVC)*, 2014.

[35] C. Ma, X. Yang, C. Zhang, and M. H. Yang. Long-term correlation tracking. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR*, pages 5388–5396, 2015.

[36] H. K. Galoogahi, T. Sim, and S. Lucey. Correlation filters with limited boundaries. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[37] H. K. Galoogahi, T. Sim, and S. Lucey. Multi-channel correlation filters. *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 3072–3079, 2013.

[38] A. V. Oppenheim, R. W. Schafer, and J. R. Buck. *Discrete-time signal processing*, volume 2. Englewood Cliffs: Prentice-hall, 1989.

[39] M. Danelljan, G. Hager, F Khan, and M. Felsberg. Discriminative scale space tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016.

[40] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg. Learning spatially regularized correlation filters for visual tracking. *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 4310–4318, 2015.

[41] C. Ma, J. B. Huang, and M. H. Yang. Hierarchical convolutional features for visual tracking. *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 3074–3082, 2015.

[42] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. *Proc. of the European Conference on Computer Vision (ECCV)*, pages 472–488, 2016.

[43] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg. Eco: Efficient convolution operators for tracking. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[44] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, and A. C. Berg. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[45] S. Hong, T. You, S. Kwak, and B. Han. Online tracking by learning discriminative saliency map with convolutional neural network. *Proc. of International Conference on Machine Learning (ICML)*, pages 597–606, 2015.

[46] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg. Convolutional features for correlation filter based visual tracking. *Proc. of IEEE International Conference on Computer Vision Workshops*, pages 58–66, 2015.

[47] L. Wang, W. Ouyang, X. Wang, and H. Lu. Stct: Sequentially training convolutional networks for visual tracking. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[48] H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4293–4302, 2016.

[49] H. Nam, M. Baek, and B. Han. Modeling and propagating cnns in a tree structure for visual tracking. *arXiv preprint arXiv:1608.07242*, 2016.

[50] H. Fan and H. Ling. Sanet: Structure-aware network for visual tracking. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[51] R. Tao, E. Gavves, and A. W. Smeulders. Siamese instance search for tracking. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[52] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr. Fully-convolutional siamese networks for object tracking. *Proc. of the European Conference on Computer Vision Workshops*, pages 850–865, 2016.

[53] J. Valmadre, L. Bertinetto, J. F. Henriques, A. Vedaldi, and P. H. Torr. End-to-end representation learning for correlation filter based tracking. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[54] Z. Hong, C. Wang, X. Mei, D. Prokhorov, and D. Tao. Tracking using multilevel quantizations. *Proc. of the European Conference on Computer Vision (ECCV)*, pages 155–171, 2014.

[55] J. Zhang, S. Ma, and S. Sclaroff. Meem: Robust tracking via multiple experts using entropy minimization. *Proc. of the European Conference on Computer Vision (ECCV)*, pages 188–203, 2014.

[56] F. Mokhtarian, S. Abbasi, and J. Kittler. Efficient and robust retrieval by shape content through curvature scale space. *Series on Software Engineering and Knowledge Engineering*, 8:51–58, 1997.

[57] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4): 509–522, 2002.

[58] H. Ling and D.W. Jacobs. Shape classification using the inner-distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(2):286–299, 2007.

[59] F. Mokhtarian and S. Abbasi. Shape similarity retrieval under affine transforms. *Pattern Recognition*, 35(1):31–41, 2002.

[60] G. McNeill and S. Vijayakumar. Hierarchical procrustes matching for shape retrieval. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 885–894, 2006.

[61] P. F. Felzenszwalb and J. D. Schwartz. Hierarchical matching of deformable shapes. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 885–894, 2007.

[62] T. Adamek and N. E. O'Connor. A multiscale representation method for nonrigid shapes with a single closed contour. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(5):742–753, 2004.

[63] E. Attalla and P. Siy. Robust shape similarity retrieval based on contour segmentation polygonal multiresolution and elastic matching. *Pattern Recognition*, 38(12):2229–2241, 2005.

[64] X. Shu and X. J. Wu. A novel contour descriptor for 2d shape matching and its application to image retrieval. *Image and vision Computing*, 29(4):286–294, 2011.

[65] C. Xu, J. Liu, and X. Tang. 2d shape matching by contour flexibility. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(1):180–186, 2009.

[66] W. Y. Kim and Y. S. Kim. A region-based shape descriptor using zernike moments. *Signal Processing: Image Communication*, 16(1):95–102, 2000.

[67] Žunić Joviša, Kaoru Hirota, and P. L. Rosin. A hu moment invariant as a shape circularity measure. *Pattern Recognition*, 43(1):47–57, 2010.

[68] Van Nguyen Hien and Fatih Porikli. Support vector shape: A classifier-based shape representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(4): 970–982, 2013.

[69] X. Bai, X. Yang, L. J. Latecki, W. Liu, and Z. Tu. Learning context-sensitive shape similarity by graph transduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(5):861–874, 2010.

[70] X. Yang, S. Koknar-Tezel, and L. J. Latecki. Locally constrained diffusion process on locally densified distance spaces with applications to shape retrieval. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 357–364, 2009.

[71] A. Temlyakov, B. C. Munsell, J. W. Waggoner, and S. Wang. Two perceptually motivated strategies for shape classification. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2289–2296, 2010.

[72] G. A. Gescheider. *Psychophysics: the fundamentals*. 2013.

[73] A. K. Jain. *Fundamentals of digital image processing*. 1989.

[74] H. Qi, K. Li, Y. Shen, and W. Qu. An effective solution for trademark image retrieval by combining shape description and feature matching. *Pattern Recognition*, 43(6): 2017–2027, 2010.

[75] J. Chen, S. Shan, C. He, G. Zhao, M. Pietikainen, X. Chen, and W. Gao. Wld: A robust local image descriptor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1705–1720, 2010.

[76] L. C. Molina, L. Belanche, and À. Nebot. Feature selection algorithms: A survey and experimental evaluation. *Proc. of IEEE International Conference on Data Mining (ICDM)*, pages 306–313, 2002.

[77] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[78] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1:886–893, 2005.

[79] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *Proc. of International Conference on Learning Representations (ICLR)*, 2015.

[80] R. C. Gonzalez and R. E. Woods. Digital image processing. *Prentice Hall, New Jersey*, 2008.

[81] C. Aslan, A. Erdem, E. Erdem, and S. Tari. Disconnected skeleton: Shape at its absolute scale. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30 (12):2188–2203, 2008.

[82] M. R. Daliri and V. Torre. Robust symbolic representation for shape recognition and retrieval. *Pattern Recognition*, 41(5):1782–1798, 2008.

[83] A. Peter, A. Rangarajan, and J. Ho. Shape l'ane rouge: Sliding wavelets for indexing and retrieval. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.

[84] I. C. Paula, F. N. S. Medeiros, and F. N. Bezerra. Shape retrieval by corners and dynamic space warping. *Proc. of International Conference on Digital Signal Processing*, pages 1–6, 2013.

[85] Z. Tu and A. L. Yuille. Shape matching and recognition-using generative models and informative features. *Proc. of the European Conference on Computer Vision (ECCV)*, pages 195–209, 2004.

[86] Y. Zhou, J. Wang, Q. Zhou, X. Bai, and W. Liu. Shape matching using points co-occurrence pattern. *Proc. of International Conference on Image and Graphics*, pages 344–349, 2011.

[87] Z. Wang and J. Ouyang. Shape classes registration and retrieval based on shape parts matching. *Journal of Computational Information Systems*, 9(4):1493–1499, 2013.

[88] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[89] P. Viola and M. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[90] G. Nebehay and R. Pflugfelder. Clustering of static-adaptive correspondences for deformable object tracking. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2784–2791, 2015.

[91] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao. Multi-store tracker (muster): a cognitive psychology inspired approach to object tracking. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 749–758, 2015.

[92] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[93] Y. Hua, K. Alahari, and C. Schmid. Online object tracking with proposal selection. *Proc. of IEEE International Conference on Computer Vision(ICCV)*, pages 3092–3100, 2015.

[94] J. Van De Weijer, C. Schmid, J. Verbeek, and D. Larlus. Learning color names for real-world applications. *IEEE Transactions on Image Processing*, 18(7):1512–1523, 2009.

[95] B. Kulis. Metric learning: A survey. *Foundations and Trends in Machine Learning*, 5 (4):287–364, 2012.

[96] N. Jiang, W. Liu, and Y. Wu. Learning adaptive metric for robust visual tracking. *IEEE Transactions on Image Processing*, 20(8):2288–2300, 2012.

[97] G. Tsagkatakis and A. Savakis. Online distance metric learning for object tracking. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(12):1810–1821, 2011.

[98] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. *Advances in Neural Information Processing Systems (NIPS)*, pages 505–512, 2003.

[99] L. Yang, R. Jin, and R. Sukthanar. Bayesian active distance metric learning. *Proc. of Uncertainty in Artificial Intelligence (UAI)*, pages 442–229, 2012.

[100] X. Liu and T. Yu. Gradient feature selection for online boosting. *Proc. of IEEE International Conference on Computer Vision (ICCV)*, 2007.

[101] Z. Kalal, J. Matas, and K. Mikolajczyk. Pn learning: Bootstrapping binary classifiers by structural constraints. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 49–56, 2010.

[102] K. Zhang, L. Zhang, and M. H. Yang. Real-time object tracking via online discriminative feature selection. *IEEE Transactions on Image Processing*, 22(12):4664–4677, 2013.

[103] A. Philipov and M. E Glickman. Multivariate stochastic volatility via wishart processes. *Journal of Business & Economic Statistics*, 24(3):313–328, 2006.

[104] A. G. Wilson and Z. Ghahramani. Generalised wishart processes. *Proc. of Uncertainty in Artificial Intelligence (UAI)*, pages 736–744, 2010.

[105] A. Smith, A. Doucet, N. D Freitas, and N. Gordon. Sequential monte carlo methods in practice. *Springer Science & Business Media*, pages 79–86, 2013.

[106] A. Bosch and A. Zisserman. Image classification using random forests and ferns. *Proc. of IEEE International Conference on Computer Vision(ICCV)*, 2007.

[107] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1794–1801, 2009.

[108] D. Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *Journal of computer and System Sciences*, 66(4):671–687, 2003.

[109] K. P. Murphy. Machine learning: a probabilistic perspective. *MIT press*, pages 848–855, 2012.

[110] T. B Dinh, N. Vo, and G. Medioni. Context tracker: Exploring supporters and distracters in unconstrained environments. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1177–1184, 2011.

[111] N. Wang, S. Li, A. Gupta, and D. Y. Yeung. Transferring rich feature hierarchies for robust visual tracking. *arXiv preprint arXiv:1501.04587*, 2015.

[112] Y. Li and J. Zhu. A scale adaptive kernel correlation filter tracker with feature integration. *Proc. of the European Conference on Computer Vision Workshops*, pages 254–265, 2014.

[113] X. Zhu, J. Lafferty, and R. Rosenfeld. Semi-supervised learning with graphs. *Doctoral Dissertation. Carnegie Mellon University, CMU-–LTI-–05-–192*, 2005.

[114] Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. *Advances in Neural Information Processing Systems (NIPS)*, pages 529–536, 2004.

[115] X. Zhu and A. B. Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.

[116] M. Kristan, R. Pflugfelder, R. Leonardis, et al. The visual object tracking vot2014 challenge results. *Proc. of the European Conference on Computer Vision Workshops*, pages 191–217, 2014.

[117] V. N. Boddeti, T. Kanade, and B. V. K. Kumar. Correlation filters for object alignment. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2291–2298, 2013.

[118] R. M. Gray. Toeplitz and circulant matrices: A review. *Foundations & Trends® in Communications & Information Theory*, 2006.

[119] M. Combescure. Block-circulant matrices with circulant blocks, weil sums, and mutually unbiased bases. *Journal of Mathematical Physics*, 50(3), 2009.

[120] Y. Xu, W. Yin, and S. Osher. Learning circulant sensing kernels. *Rice University, Department of Computational and Applied Mathematics*, 2014.

[121] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. Pflugfelder, G. Fernandez, G. Nebehay, F. Porikli, and L. Cehovin. A novel performance evaluation methodology for single-target trackers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016.

[122] M. Kristan, R. Pflugfelder, A. Leonardis, et al. The visual object tracking vot2013 challenge results. *Proc. of IEEE International Conference on Computer Vision Workshops*, pages 98–111, 2013.

[123] Z. Wang and S. Vucetic. Online training on a budget of support vector machines using twin prototypes. *Statistical Analysis and Data Mining*, 3(3):149–169, 2010.

[124] Y. Wu, J. Lim, and M. H. Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1834–1848, 2015.

[125] G. Zhu, J. Wang, C. Zhao, and H. Lu. Weighted part context learning for visual tracking. *IEEE Transactions on Image Processing*, 24(12):5140–5151, 2015.

[126] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. Torr. Staple: Complementary learners for real-time tracking. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[127] K. He, X. Ren, and J. Sun. Deep residual learning for image recognition. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[128] N. Wang and D. Y. Yeung. Learning a deep compact image representation for visual tracking. *Advances in Neural Information Processing Systems (NIPS)*, pages 809–817, 2013.

[129] K. Zhang, Q. Liu, Y. Wu, and M. H. Yang. Robust visual tracking via convolutional networks without training. *IEEE Transactions on Image Processing*, 25(4):1779–1792, 2016.

[130] B. Cai, X. Xu, K. Xing, K. Jia, J. Miao, and D. Tao. Bit: Biologically inspired tracker. *IEEE Transactions on Image Processing*, 25(3):1327–1339, 2016.

[131] D. Y. Lee, J. Y. Sim, and C. S. Kim. Multihypothesis trajectory analysis for robust visual tracking. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5088–5096, 2015.

[132] P. F. Felzenszwalb and R. Zabih. Dynamic programming and graph algorithms in computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33 (4):721–740, 2011.

[133] H. Ishikawa. Exact optimization for markov random fields with convex priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1333–1336, 2003.

[134] A. Vedaldi and K. Lenc. Matconvnet: Convolutional neural networks for matlab. *Proc. of ACM International Conference on Multimedia*, pages 689–692, 2015.

[135] J. Gao, H. Ling, W. Hu, and J. Xing. Transfer learning based visual tracking with gaussian processes regression. *Proc. of the European Conference on Computer Vision (ECCV)*, pages 188–203, 2014.